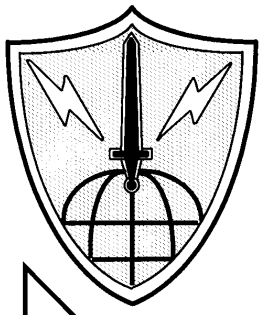
TR NO. AMSEL-IE-EA-98142



COMMAND AND CONTROL (C2)

PROTECT TOOL KIT

IMPLEMENTATION GUIDANCE

FOR

UNIX OPERATING SYSTEM

USAISEC

JUNE 1998

SPECIAL PROJECT OFFICE
AMC EXECUTIVE AGENT FOR CORPORATE INFORMATION

DEPARTMENT OF THE ARMY
U.S. ARMY INFORMATION SYSTEMS ENGINEERING COMMAND
FORT HUACHUCA, ARIZONA 58613-5300

DISCLAIMER

The findings of this document are not to be construed as an official Department of the Army position unless designated by other authorized documents. The use of trade names in this document does not constitute an official endorsement or approval of the use of such commercial hardware or software.

CHANGES

Send all changes which affect this document to the Commander, U.S. Army Information Systems Engineering Command, ATTN: AMSEL-IE-SP, Fort Huachuca, AZ 85613-5300.

DISPOSITION INSTRUCTIONS

This document will be destroyed when no longer needed (do not return to the originator). Safeguarding and destruction of this document will be performed with consideration of its limited distribution requirements.

TR No. AMSEL-IE-EA-98142
June 1998

C2 Protect Tool Kit Implementation Guidance for UNIX Operating Systems

TABLE OF CONTENT

Paragraph **Page**

APPENDIX

FIGURES

FIGURES

C2 Protect Tool Kit Implementation Guidance for UNIX Operating Systems

1.0 Introduction.

1.1 Purpose.

The purpose of this document is to aid UNIX System Administrators in the acquisition, installation, configuration and usage of the three mandated Command and Control (C2) Protect Tools. These tools are Security Profile Inspector for Networks (SPI-NET), Transmission Control Protocol (TCP) Wrapper, and Simple Watcher (swatch). It is intended to provide sufficient information and instruction that will enable the System Administrator to easily and confidently, load and operate the C2 Protect Tools and thus obtain a strengthened system security posture.

1.2 Scope.

This document and guidance only applies to UNIX operating systems and is intended solely as guidance to aid System Administrators in performing the tasks associated with acquiring, installing, configuring, and operating the C2 Protect Tools. System administrators are ultimately responsible for the proper configuration and usage of the C2 Protect Tools.

1.3 Background.

The C2 Protect Tool Kit was introduced through a Director of Information Systems for Command, Control, Communications, and Computers (DISC4) memorandum, dated 11 Apr 96. The memorandum states,

> "The Army C2 Protect Program Implementation Plan and AR 380-19 (Information Systems Security) requires that all prudent measures be taken to protect our systems; the information they manage, produce, store and distribute. The Army C2 Protect library defines the need for the acquisition, integration and implementation of C2 Protect software applications into information systems by System Administrators and managers. As a result of recent intrusions and penetrations or Army systems by unauthorized elements, network managers in all environments are directed to acquire C2 Protect Tools for use in the security management of their systems. System developers are advised to integrated GFE security protection, detection and misuse tools during their system design and build phases. These tools will aid in this effort."

This initial C2 Protect Tool Kit memorandum identified Security Profile Inspector (SPI), TCP Wrapper and swatch as mandated security tools to be used on all UNIX operating system platforms. As time has progressed, the TCP Wrapper and swatch tools have remained a vital component of the C2 Protect Tool Kit, however support for SPI has been redirected to an improved applications called SPI-NET. Consequently, the focus of the document will be on
SPI-NET rather than the original SPI.

2.0 General Information.

2.1 C2 Protect Tool Kit Information.

The Defense Information Systems Activity (DISA) Automated Systems Security Incident Support Team (ASSIST) and the Army Computer Emergency Response Team (ACERT) have been designated as primary support organizations. The ASSIST has the following mission statement:

> "The Automated Systems Security Incident Support Team (ASSIST) is the INFOSEC Incident Response Support to the Defense Information Infrastructure (DII) Community in Support of Information Assurance.
>
> Our Goal is to Identify, Analyze, Assess and Resolve all INFOSEC Vulnerabilities and Exploitations in the DII in Support of the Defense Information Systems Agency's (DISA) Information Assurance Mission."

The mission statement for the ACERT is as follows:

> "The Army Computer Emergency Response Team (ACERT) conducts command control protect operations in support of the Army to ensure the availability, integrity, and confidentiality of the information and information systems used in planning, directing, coordinating, and controlling forces in the accomplishment of the mission across the full spectrum of support to military operations."

Both ASSIST and ACERT provide a variety of computer security services and resources. Most important among these services and resources are security bulletins, frequently asked questions (FAQs), links to vendor security patches, incident reporting information, and access to the C2 Protect Tool Kit.

Lawrence Livermore National Laboratories (LLNL) is the developer of the SPI-NET component of the C2 Protect Tool Kit. They also developed the original SPI and SPI for NT. Unlike the TCP Wrapper and swatch, SPI-NET is an U.S. Government controlled software item.

2.2 C2 Protect Tool Kit Download Sites.

The C2 Protect Tools are available to download from several file transfer protocol (FTP) areas and websites. In order to prevent the repetitious definition of the sites, as part of each of the C2 Protect Tools installation procedures, the following is a list of sites associated with the C2 Protect Tools:

ASSIST:	WWW	www.assist.mil
	FTP	ftp.assist.mil/pub/tools/
ACERT	WWW	www.acert.belvoir.army.mil/webtools.html
	FTP	ftp.acert.belvoir.army.mil/pub/unix.toolbox/
LLNL	WWW	ciac.llnl.gov/cstc/cstc.html

2.3 Additional C2 Protect Tools.

Recently, additional security tools have been added to the UNIX C2 Protect Tool Kit. However, these tools are not yet mandated for use on all systems but have been added as additional resources to aid the System Administrator effectively securing their system. The additional tools and short synopsis of each tool are found in Appendix B: Additional UNIX C2 Protect Tools.

2.4 Document Legend.

To aid the reader in discerning what is occurring on the computer system and what is the textual guidance, different fonts will be used to differentiate between textual guidance, keyboard actions, and computer configuration files.

All the text, printed using the Arial font, are UNIX command text and designed to be typed at the command line.

```
All text, printed using the Courier New font, are UNIX system
file readings and are designed to show portions of
configuration files or other text information located on the
system.
```

3.0 Security Profile Inspector for Networks (SPI-NET).

The SPI-NET is a multifaceted security program developed by LLNL for use by the U.S. Government. SPI-NET consists of six tools that assist UNIX System Administrators in maintaining systems security and detecting system vulnerabilities.

3.1. Basic Features.

SPI-NET, a marked improvement over the original SPI, includes a remote operations (network) capability, and all configuration and operational functions are performed via a graphical user interface (GUI). As mentioned above, SPI-NET consists of six security tools. These tools are:

1) Access Control Test (ACT). The ACT examines sequential dependencies in UNIX access control files and permissions to attempt to obtain a goal (target) using a given initial condition.

2) Binary Authentication Tool (BAT). This tool checks the authenticity of system objects against MD5 checksums provided by the vendor.

3) Change Detection Test (CDT). CDT is used to track changes to important system files, user/group accounts and related system attributes. This is probably the most important tool in SPI-NET.

4) Configuration Query Language (CQL). CQL gives the System Administrator the ability to look for bad system configurations based on system and applications specific parameters.

5) Password Security Inspection (PSI). The PSI tool checks the system password file for weak user password entries.

6) Quick System Profile (QSP). QSP check for system security vulnerabilities using a standard set of tests to test for operating system vulnerabilities.

3.2 Operating Concepts.

Before discussing the steps required to install, configure, and operate SPI-NET, a quick explanation of the basic SPI-NET operating concepts is essential. SPI-NET can be operated in either a single-host (standalone) or multi-host environment. In order to support the multi-host environment, SPI-NET was designed with two separate software components. These components are the “command console” and the “engine” portions. The engine does all the actual work while the command console allows the operator to perform the management and reporting functions. For a single host, or standalone implementation, both the command console and the engine portions are loaded on the system. To implement a multi-host configuration, the engine software must be loaded on all the systems, while the command console code is loaded on a single machine. From this single command console equipped machine (command host), the

System Administrator can configure and operate the SPI-NET security tools on all of the machines. To support this remote capability in a secure manner, command and data traffic between the remote engines and the command host is protected by public key authentication encryption techniques.

3.3 Obtaining and Installing SPI-NET.

3.3.1 Software Packages.

The SPI-NET code is available in a variety of packages from ASSIST, ACERT or LLNL. The code is available for either the standalone implementation or detached (remote) environments. The standalone implementation includes both the command console and the engine components while the detached or remote implementation consists of only the engine software code. The code packages themselves are offered in either a source code or binary code versions. Binary code versions are available for HP-UX 10.X, IRIX 5.X, SunOS 4.X (((SPARC) Only) and Solaris 2.X (SPARC Only)).

The latest version of the SPI-NET code is version 1.01. For the source code packages, the distribution is identified as spinS-1.01.tar.Z.des (standalone) and spinRS-1.01.tar.Z.des (remote). The binary code version is identified as spin.b-1.01.<OS Type>.tar.Z.des (standalone) and spin.rb-1.01.<OS Type>.tar.Z.des (remote).

3.3.2 Prerequisites.

Since the distribution of SPI-NET is limited to U.S. Government entities, procedures are in place to control the distribution and usage of the code packages. First, to control the distribution of the software, ACERT and ASSIST will only allow hosts having a Domain Name Server (DNS) registered ".mil" host name to download the software packages. In order to download the code from LLNL, a download password must first be obtained via either e-mail (.mil addresses only) or telephonically from LLNL. This procedure effectively limits the distribution of the code to only U.S. Government entities. However, even with the code in hand, a data encryption standard (DES) decryption key is still need to open up and access the SPI-NET code package. The DES decryption key can be obtained via e-mail (.mil addresses only) or telephonically (Defense Switched Network (DSN) numbers only). If your machine does not have the DES software loaded it can be obtained from ASSIST or ACERT and is identified by the package name libdes-3.23.tar.Z.

3.3.3 Installing SPI-NET.

For the purposes of this guide, we will discuss the procedures associated with installing the standalone source code package. This is the hardest package to install to accomplish and should provide key insights in the installation of the remote source code package and the binary code packages. The following are basic steps to install the standalone source code package.

1. Download the SPI-NET source code file (spinS-.1.01.tar.Z.des) from ASSIST's FTP site or other suitable location. Place the file in the directory under which you want this tool to be created. Note that SPI-NET will create its own subdirectories for the installation and the installation will be permanent. For the purposes of this guide we will use a sample subdirectory, /secure, for the basis of the installation.

http://www.assist.mil OR
ftp://ftp.assist.mil/pub/tools/spi-net/

If not previously acquired, obtain the decryption key from ASSIST via phone or email request. ASSIST will email the decryption key to an address in the .mil domain.

2. Decrypt the SPI-NET source code file. Note that DES software must be loaded on the machine.

```
des -d - k XXXXXX <spi[ ... ].tar.Z.des > spi[...].tar.Z
        (DES KEY)
                (spi file name)
                                (spi file name)
```

Example: des -d -k XXXX <spinS-1.01.tar.Z.des> spinS-1.01.tar.Z

3. Uncompress SPI-NET using the following command:

```
uncompress spinS-1.01.tar.Z
```

4. Extract SPI-NET source code using the following command:

tar -tvf spinS-1.01.tar	view the contents of the tar file
tar -xvf spinS-1.01.tar	extract the contents of the tar file

5. Read the README file.

```
more README
```

6. The actual installation command used to install SPI-NET depends on the installation package that being utilized. Since this example uses the standalone source code package, the following command is used to start the installation.

```
./Install
```

For the other installation packages, different installation commands are used. For the installation of the remote source code package use "./Installr"; while for either of the binary code packages use the installation command "./Setup".

As part of the installation process, the operator will be asked several questions in order to complete the installation. The operator will be prompted to provide:

- System Hostname

- C Compiler name/location

- additional compiler and linker flags

- Develop distribution entity (Y/N). A distribution entity allows the system to build installed versions for other platforms.

- Command Host address.

- Host ID (remote installation only)

- Take Change Detection Test (CDT) snapshot (Y/N)

- Remove SPI-NET source code (Y/N)

3.4 Getting Started.

3.4.1 Starting SPI-NET.

As was described previously, the SPI-NET code consists of two separate and distinct software components. These components are the engine or remote agent and the command host or console. In a standalone implementation, both of these components must be installed and operating in order for SPI-NET for work. In a remote implementation, only the remote agent is required. In order to prepare SPI-NET for execution, various services must be started. First, for all hosts that are to be managed, the Remote Communications Server (RCS) must be started. This is the remote agent or engine needed to receive process instructions from the command console, perform the actual security jobs, and report the results back to the command host. The command host has two services that are associated with the command console function. These services are the Master Communications Server (MCS) and the Job Control System (JCS). The MCS controls all command host to remote host communications and the JCS controls the scheduling and execution of the SPI-NET security tools.

As previously stated, in order to operate SPI-NET the RCS must be started on the command host and any target machine that is being managed by the SPI-NET command host. To start the RCS, perform the following:

```
cd /secure/spin-1.01/binr/
./StartRCS
```

NOTE: All remote services and actions are located in the …/spin-1.01/*binr*/ directory.

To activate the command host, the MCS and the JCS must be activated. The activation of the MCS and JCS can be performed via the command or through the SPI-NET graphical interface. Usage of the graphical interface is the method normally used. If the command line is used, perform the following commands:

```
cd /secure/spin-1.01/binm/
./StartMCS
```

./StartJCS

NOTE: All command host services and actions are located in the …/spin-1.01/*binm*/ directory.

The SPI-NET graphical interface is started and used separately from the RCS, MCS, and JCS services. Additionally, if the RCS, MCS, and JCS are activated, SPI-NET will perform the scheduled activity regardless of whether the SPI-NET graphical interface is running or not. The SPI-NET graphical interface allows the operator to manage SPI-NET security activities, perform immediate jobs, and view reports. To start the SPI-NET graphical interface, perform the following commands:

```
cd /secure/spin-1.01/binm/
./spinet
```

On start-up of SPI-NET the operator is given the option to start MCS and JCS if they are not active. To start MCS and JCS from the menu system use `Status-> Start System Process -> Master Com Server` and `Status-> Start System Process -> Job Control System`. See Figure 3-1 for a view of the SPI-NET menu bar.

3-1. SPI-NET Menu Bar

At the first time start-up, you will be given the option of establishing a password. This password provides access control to the management of SPI-NET, and the viewing of the security reports. The password can also be changed via the menu using `File ->SPI-NET Login/Logout`. 3-2 shows the initial password entry screen.

Figure 3-2. Password Entry Screen.

Display options are available that allow the main menu to show a status bar and/or the Report Listing. These options are available through `Options -> Toggle Report Listing` and `Options -> Toggle Status Bar`. The Status Bar, as seen in Figure 3-3, displays (from the left) the JCS status, the current time, the time of the next scheduled event, and the MCS status. Figure 3-4 shows a sample menu bar with the Status Bar and the Report Listing.

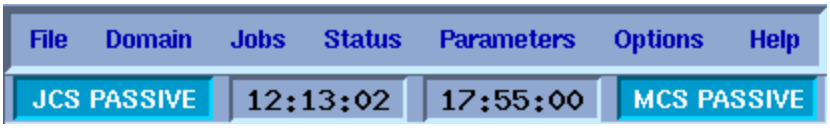


Figure 3-3. Status Bar.

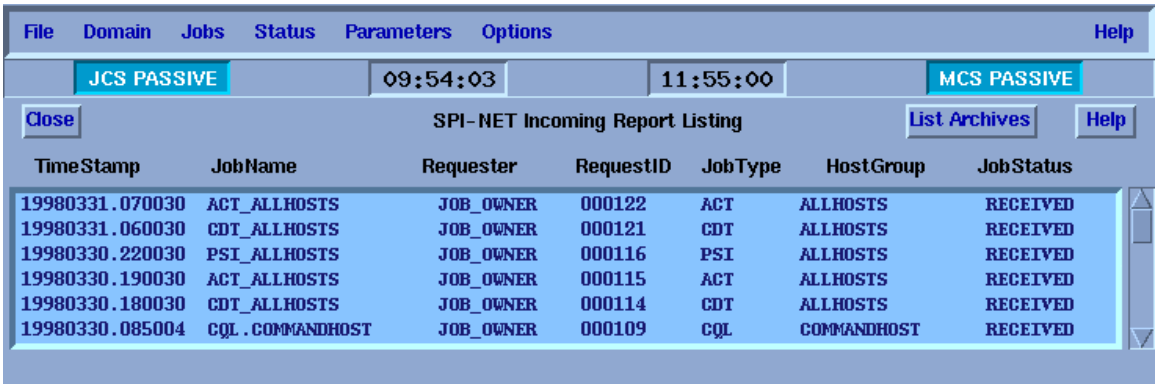


Figure 3-4. Report Listing.

3.4.2 Checking SPI-NET Operations.
To check that SPI-NET is fully operational, perform the following items:

1. Ensure that the JCS and MCS are started by checking the status bar or using `Status -> Test System Status -> Job Control System` and `Status -> Test System Status -> Master COM Server.`

2. Ensure that the RCS is started on the appropriate host(s). This is done by checking the process status of the system looking for a "rcsx" entry, use the command “ps -elf | grep rcsx”.

NOTE: Only one instance of the RCS should be running at any one time on any given system. If multiple instances are running, stop all instances of the RCS and start only a single instance of RCS.

3. Test the communications between the command host and the remote agent(s) using `Status -> Test Remote Com Servers`. Remember; even in a standalone implementation, a command host and a remote agent are utilized. Figure 3-5 shows the Remote Communication Server Testing. To execute the test first, select a HostGroup from `Target Host Group` and then press `Start Test.`

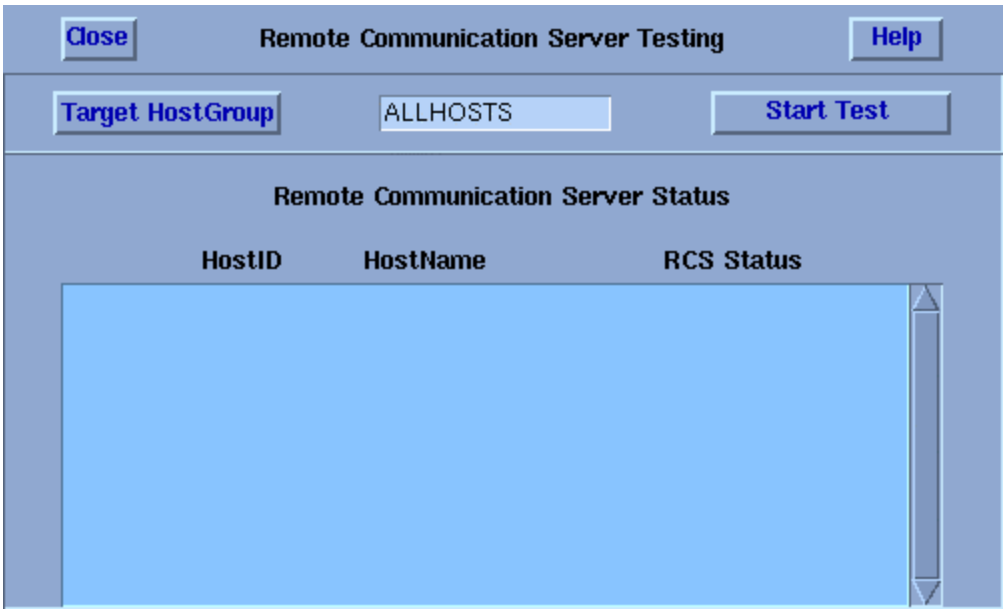


Figure 3-5. Remote Communications Server Testing.

The command host will try to communicate with the remote agent and will actively report the status in the RCS Status column. If the host is not responding, click on `Help` for more information about SPI-NET communications. See Figure 3-6 for an example of a remote communications testing session.

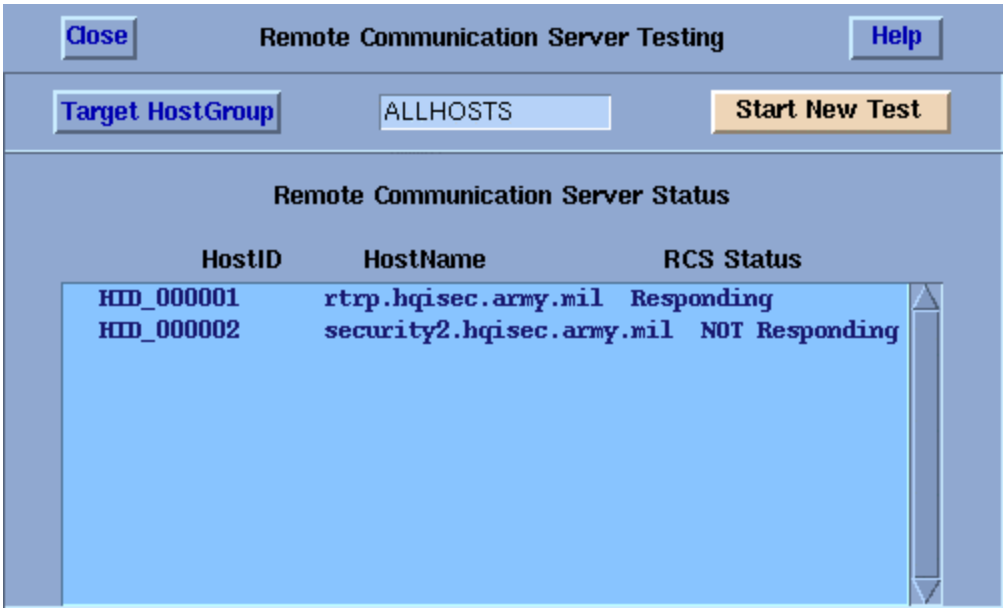


Figure 3-6. Remote Communications Test Example.

3.4.3 SPI-NET Parameter File Basics.

SPI-NET uses two types of parameter files. These files are pre-loaded templates and machine templates. Pre-loaded templates, loaded during the installation process, are used as generic operational templates and to build machine templates. The pre-loaded templates are read-only and have no

extension associated with them. Machine templates are parameter templates tailored, by the operator, to each system and are identified by a HostID name extension.

Each of the SPI-NET security tools have three configuration levels (1, 2, and 3), except for the CDT, which also has a Level 0 configuration for baseline snapshots. By default, Level 1 is the least intense (less secure) and Level 3 is the more intense (more secure) of the configuration levels. The Level parameter for all the tools are adjustable, by the operator, to meet specific needs and operating scenarios.

Parameter templates for the SPI-NET tools are accessed from the menu by `Parameters -> Define Inspection Parameters`. The Inspection Parameter Management screen is show in Figure 3-7. A new template can either be created from scratch or adapted from an existing template.

A template, made from scratch, is created by a four-step process.

1. Select a Host Target from the `Host Target Selection` area. This defines the host to which this machine template will be applied.
2. Specify a tool by using the `Inspector` button and associated pull-down selection area.
3. Specify a Level by using the `Level` button as associated pull-down selection area.
4. With the template defined, the parameters can be adjusted to suit the needs of the system. The tool specific parameters are located in the lower right hand portion of the Inspection Parameter Management window.

To inspector, or modify a template, first select a template (double click) from the `Parameter Template Files` area. This causes the window to display the parameters associated with that template file. The `Current Template` field reflects the tool (Inspector), level, and target (HostID) of the template currently under inspection. The syntax for the template files are <tool>_<level>.<hostid>. A simple method to generate new machine templates is to take an existing template, change the Host Target of the template, and save the template file. The configuration of specific tools is covered in Section 3.5.

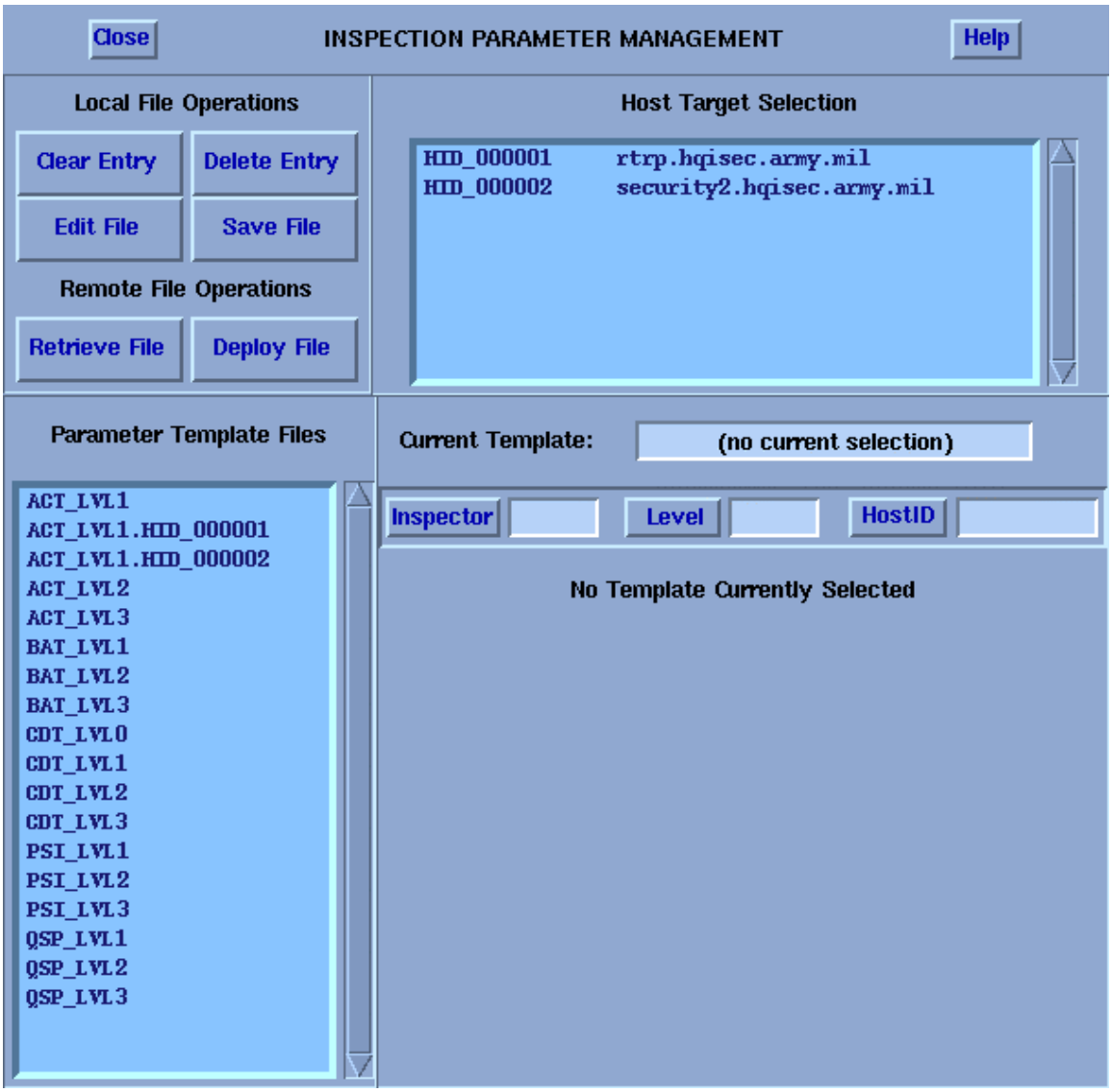


Figure 3-7. Inspection Parameter Management screen.

With the parameter template defined, the operator can Edit, Delete, or Save the parameter template as desired. All changes to new or existing template files must be saved (`Save File`) in order to be saved to file. Changes made on the screen are not saved to file by default. It is important to note that any new or modified template *must* be pushed to the host (standalone or remote) before it takes affect. This is accomplished using `Deploy File` button.

3.4.4 Executing SPI-NET Tools.

The SPI-NET security tools can be executed either by an immediate initiation or on a scheduled basis.

To start a tool immediately use `Jobs -> Immediate Launch -> Tool` where *Tool* is the SPI-NET tool that is desired. Figure 3-8 shows the Launch Immediate Tool window that is displayed when a tool is launched from the SPI-NET graphical interface.

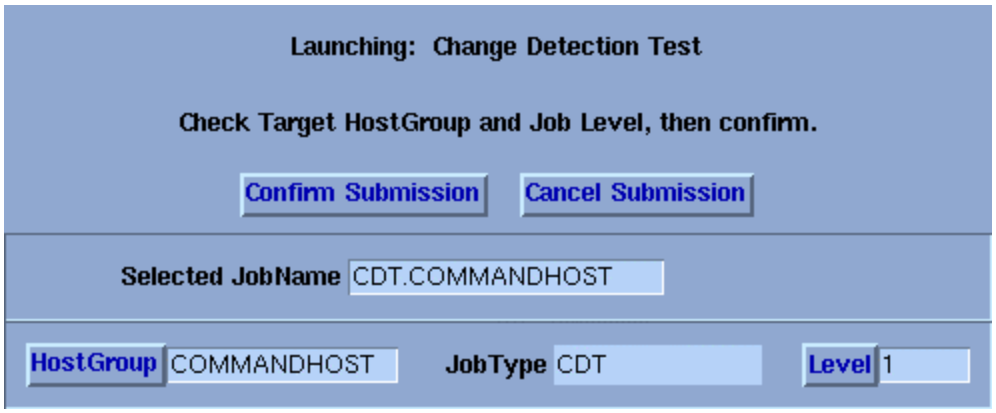


Figure 3-8. Launch Immediate Tool window.

At this window the HostGroup and Level can be adjusted as desired. The defaults for HostGroup and Level are set using `Jobs -> Immediate Launch -> Change Default Target` and `Jobs -> Immediate Launch -> Change Default Level` respectively. Use the `Cancel Submission` button to cancel this job action or the `Confirm Submission` button to begin job execution. Shortly after the start of the job, the MCS and JCS status indicators will flash and display as ACTIVE rather than PASSIVE. At this time, the Incoming Report Listing will show a new report entry. However, the JobStatus will not indicate job completion or "Received" status until the job report has been received from the remote agent by the command host. If problems occur and a proper JobStatus is not received, click on Help to receive additional job process information.

3.4.5 Scheduling Tool Execution.

In order for SPI-NET to provide constant and around the clock security services, execution of the SPI-NET security tools can be scheduled through the Job Control DataBase (JCDB). The JCDB contains JobGroup entries. These entries annotate what systems are to execute which tools at what time or interval. The JobGroup entry consists of three major parts, (1) the job type (ACT, BAT, CDT, etc), (2) the HostGroup, and (3) the inspection schedule type (single, repeat & interval). Access to the JCDB is through `Jobs -> Edit Launch Schedule`. The JCDB Management window is shown in Figure 3-9.

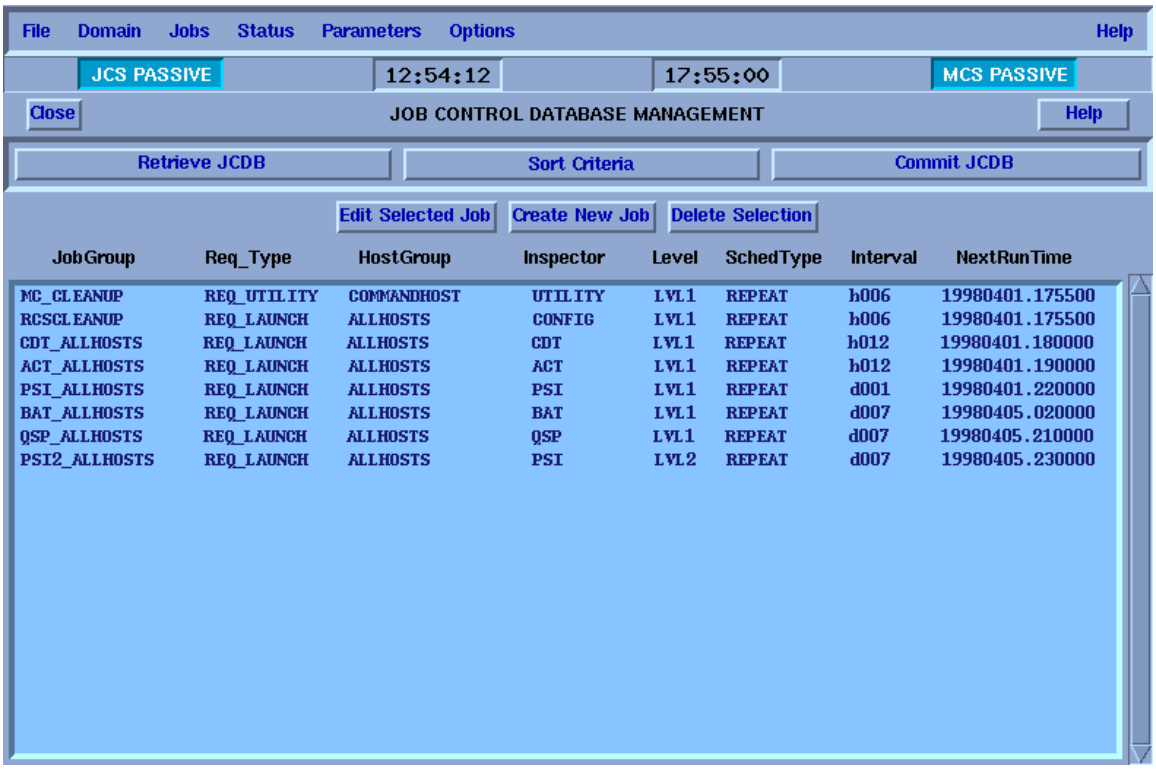


Figure 3-9. JCDB Management Window.

From the JCDB Management window, JobGroup entries can be created (`Create New Job`), edited (`Edit Selected Job`) or deleted (`Delete Selection`) by selecting the JobGroup entry and pressing the appropriate button.

Editing and creating a new JobGroup are essentially the same function and use the same editing window. Figure 3-10 shows the JobGroup entry window. Each JobGroup must a have unique name (Job Name). If a new JobGroup entry is created using an existing Job Name, SPI-NET will overwrite the existing JobGroup Entry with the new JobGroup information. New JobGroups can be created by editing an existing JobGroup and changing the Job Name, or by being created from scratch. The HostGroup, JobType, and Level define what tool will be executed on the selected HostGroup. The JobTypes include the six SPI-NET tools along with the UTILITY and CONFIG functions. UTILITY and CONFIG are clean-up functions that remove all unnecessary SPI-NET files older than a specified age (3 days). The UTILITY function applies to the command host while CONFIG applies to the remote agent portion of SPI-NET. The inspection schedule is set by using a SchedType of either SINGLE or REPEAT. For REPEAT entries, an `Interval` must be specified in days (d*nnn*), hours (h*nnn*) or minutes (m*nnn*) where *nnn* is a zero-padded three-digit number. The `Next Run Time` field indicates when the JobGroup is next scheduled to be processed. The format is YYYYMMDD.hhmmss. The insertion of the term ASAP in this field will cause the JCS to process this JobGroup at its earliest convenience. The operator via the `Submit Entry` button accepts new entries and changes to the existing JobGroups.

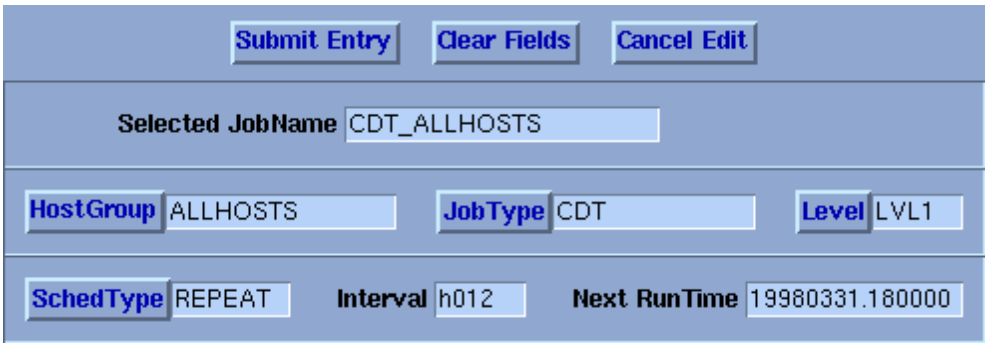


Figure 3-10. JobGroup Entry Screen.

After pressing the `Submit Entry` button, the new or modified JobGroup entry is reflected in the listing of JobGroup entries. However, in order to finalize the changes, the operator must use the `Commit JCDB` button to write the changes to the database. The `Retrieve JCDB` button retrieves the current active JCDB, and can be used to confirm that changes to the JCDB were properly written. The `Sort Criteria` button is used to adjust the JobGroup listing to sort the items on a different criteria. The JobGroup listing can be sorted by JobGroup, Inspector, HostGroup, and by Next Run Time.

3.4.6 SPI-NET Reports.

To access the SPI-NET reports, the Report Listing must be active (`Options -> Toggle Report Listing`). SPI-NET has two report listings, Incoming Reports and Archive Reports. These listings are toggled by the `List Incoming` and `List Archive` toggle button. Double clicking on a specific report, which brings up the Report Generator Display window, accesses reports. A "recent" report (List Incoming) can be archived or printed from the Report Generator Display. An archived report (List Archive) can be printed or deleted from this same window. An Archived report can also be viewed and accessed via the system by accessing the …/spin-1.01/binm/D/reps/archive/ directory.

A "patch" to SPI-NET, the autoReport script, provides a method to have reports from the Incoming Reports section sent, via electronic mail, to a designated recipient. The particulars on obtaining, installing, and using this script can be found in Appendix C: The autoReport Feature.

It should be strongly noted that, by default, reports older that three days old will be removed from the List Incoming display if they are not archived. Changes to this setting can be made by editing the MC_CLEANUP and RCSCLEANUP JobGroup entries.

3.5 Configuring & Executing the SPI-NET Tools.

Each of the six SPI-NET security tools has specific configuration parameters and usage instructions. Each tool will be discussed, and the configuration parameters explained, in the following paragraphs.

3.5.1 Access Control Test (ACT).

The ACT examines the sequential dependencies in UNIX access control files and permissions to attempt to obtain a goal (target) using a given initial condition. Using a target and initial condition set by

the operator, SPI-NET uses a rule base to determine "Can I get there from here?" Some sample goals may include:

To gain the ability to modify a specified file.

To gain the ability to execute commands with the authority of a given User.

To gain the ability to execute commands with the authority of a given Group.

As seen in Figure 3-11, the operator must set an `Initial State and a Target Goal`. A small explanation, of the syntax used, is required to properly understand these parameter entries.

Target Goal and *Initial State.* For both the Initial State and Target Goal, one of six object types must be selected. These object types identify the nature of the Initial State and Target Goal. The six-object types and associated syntax are:

UID=<number>	User Identifier
U=<user>	User Name
GID=<number>	Group Identifier
G=<group>	Group Name
FR=<file>	File Read
FW=<file>	File Write

For the Target Goal some sample goals may be:

UID=0	Achieve root equivalent status
U=root	Become root user.
GID=10	Become a member of group with GID=10
G=staff	Become a member of the staff group
FR=/etc/passwd	Achieve permission to read /etc/passwd
FW=/etc/passwd	Achieve permission to write to /etc/passwd

The default Target Goal is "UID=0". For the Initial State some sample states may be:

UID=109	User with identifier 109
U=joe	User with name "joe"
GID=32	A member of group with identifier 32
G=users	A member of group "users"
FR=/etc/passwd	Ability to read the file /etc/passwd
FW=/etc/hosts	Ability to write to file /etc/hosts

The default Initial State is "U=World"

Rule Base. The field `Rule Base` defines the location of the ACT rules file. The directory …/spin-1.01/D/data/act/ contains the transition rules that the ACT utility uses to generate intermediate subgoals in its quest for the target goal.

Exit on First Success Path ? If selected `(Yes),` ACT will report the first vulnerable path (access dependency chain) discovered and exit immediately. If No is selected, ACT will search out and report ALL vulnerable paths.

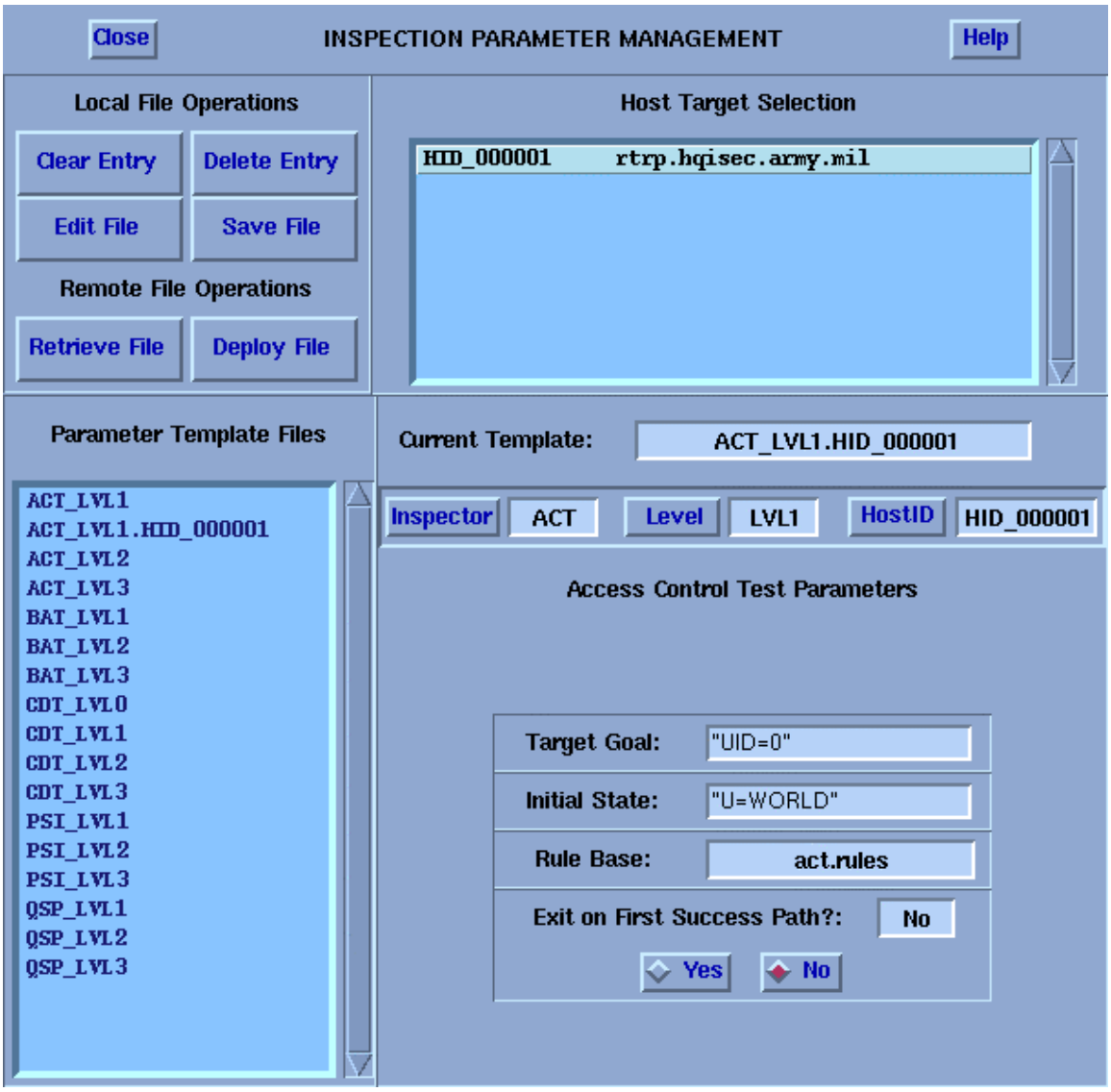


Figure 3-11. ACT Parameter Screen.

The following is a sample report that highlights the form and possible findings available from using the ACT tool. Note that the report has been edited for space and contents.

```
************* SPI-NET 1.0 ACCESS CONTROL TEST REPORT ************************

***** REPORT IDENTIFICATION SECTION *****

TARGET HOSTS REPORTING:

HostName         Start_TimeStamp    Final_TimeStamp      Elapsed_Time
--------         ---------------    ---------------      ------------
rtrp             19980401.141542    19980401.141542              00s

TARGET HOST SPECIFICATION:

HostName         OS_Type      OS_Name      OS_Version   Hardware
--------         -------      -------      ----------   --------
rtrp             UNIX         SunOS        5.5.1        i86pc

INSPECTION PARAMETER SPECIFICATION:

HostName         Inspector PathMode    RuleBase     Assumption   TargetGoal
--------         --------- --------    --------     ----------   ----------
rtrp             ACT       AllPaths    act.rules    FW=/etc/pass UID=0
                                                      wd

***** SECURITY FINDINGS SECTION *****

VULNERABLE PATHS DETECTED:

HostName          Target   TargetName   Relationship   Agent  AgentName
--------          ------   ----------   ------------   -----  ---------
rtrp              UserID   UID=0        Vulnerable_To  File   FW=/etc/pass
                                                                 wd

VULNERABLE PATHS (DETAIL):

HostName         Method
--------         ------
rtrp             ***  FW=/etc/passwd *  FR=/etc/passwd * grant U=smtp * gr
                   ant UID=0 * TARGET
rtrp             ***  FW=/etc/passwd *  FR=/etc/passwd * grant U=root * gr
                   ant UID=0 * TARGET

ACT REPORT SUMMARY:

HostName          Subject    Status       Comment
--------          -------    ------       -------
rtrp              PATHS      TOTAL_PATHS  2
```

3.5.2 Binary Authentication Tool (BAT).

The BAT checks the authenticity of system objects against a Message Digest 5 (MD5) checksum of those objects. The vendor provides the MD5. However, there is limited applicability. The checksum

tables have not recently been updated to account for the newer operating systems. Currently support is limited to:

IRIX 5.2/5.3
SunOS 4.1.2 - 4.1.4
SunOS 5.3/5.4
Ultrix 4.4

3.5.3 Change Detection Tool (CDT).

The Change Detection Tool, probably the most important tool offered by SPI-NET, is used to track changes to important system files, user/group accounts, and related system attributes. The purpose of this tracking is twofold: to help detect intrusions and to warn the System Administrator when the attributes of system critical files have been inadvertently modified during routine administrative actions. The CDT can be used as a simple intrusion detection system when used in a regular manner. When the CDT tool is run it compares the current file system against a previously generated baseline snapshot of the file system. A baseline snapshot can be performed at installation time or by running a Level 0 CDT. A Level 0 snapshot must be created before running CDT Levels 1-3. As can be seen in Figure 3-12, the two major file specifications are the CDT MetaSpec File and the CDT DataBase File. The CDT MetaSpec File contains specific information about what is to be checked on the file system. The CDT DataBase File contains the baseline data against which a future CDT comparison will be performed. It should be noted that a new snapshot would overwrite any previously created baseline snapshot for any particular MetaSpec File and DataBase File. Multiple MetaSpec/database file pairs can be used on a system to help divide the workload.

NOTE: For any CDT template used by a system, the MetaSpec file and the database file must be a matched pair.

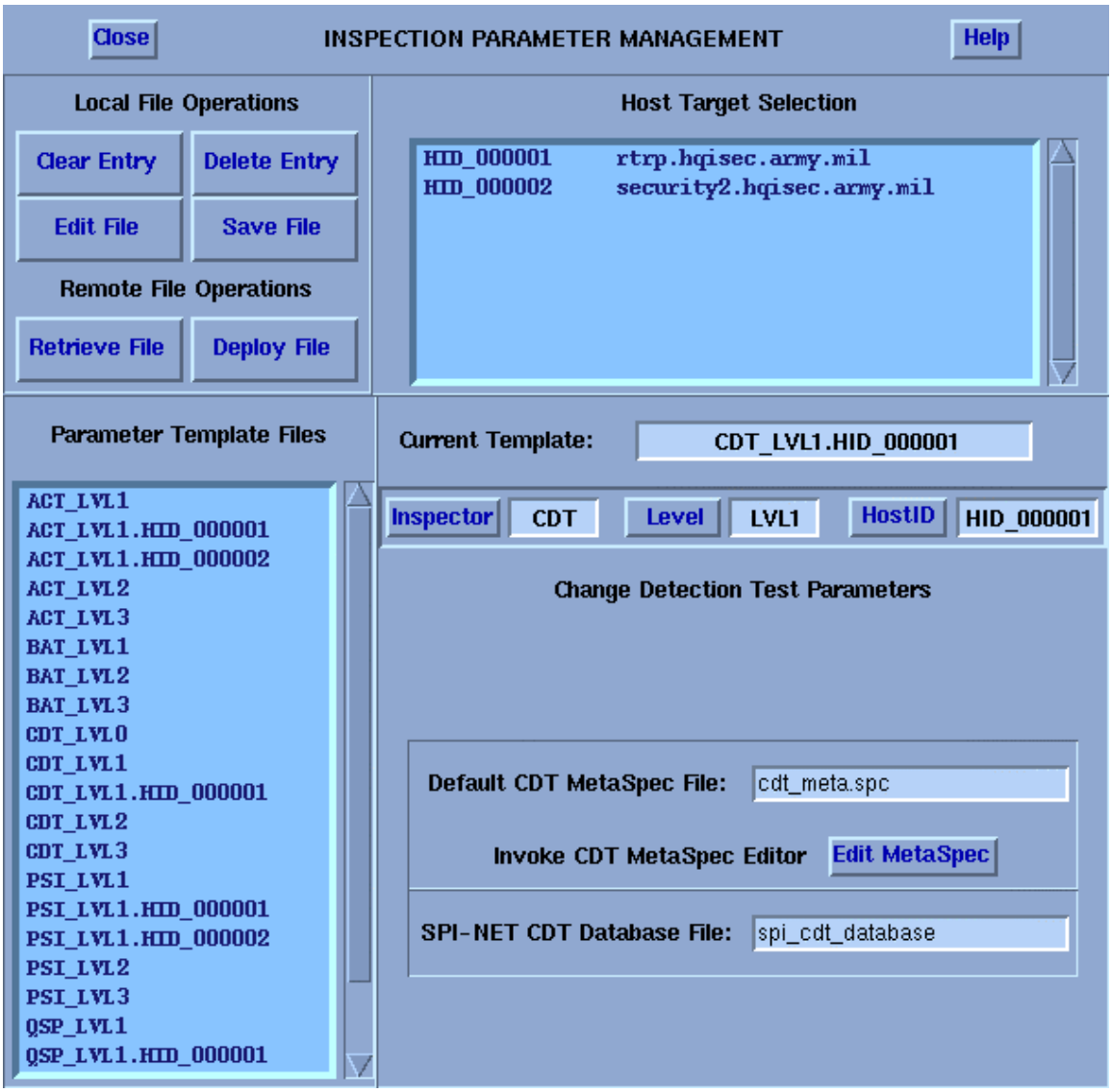


Figure 3-12. CDT Parameter Screen.

The heart of the CDT configuration is located in the CDT MetaSpec File and can be accessed through the MetaSpec Editor (`Edit MetaSpec`). Figure 3-13 shows the basic layout of the CDT MetaSpec Editor. The CDT MetaSpec Editor consists of three primary areas. From the top these areas are:

1) MetaSpec File & Target Host Selection area. In this area the Target host is selected and the MetaSpec file identified. When an existing MetaSpec File is selected, the contents of that file are represented in the MetaSpec Entry Selection area.

2) MetaSpec Entry Selection area. This area displays the current MetaSpec entries for the selected MetaSpec file. Selection of specific entry lines, will display the entry specifics in the MetaSpec Entry Editing area.

3) MetaSpec Entry Editing area. This is an area where the line entries, used in the MetaSpec, can be created, modified, or deleted. It is in this area that the actual MetaSpec data is defined.

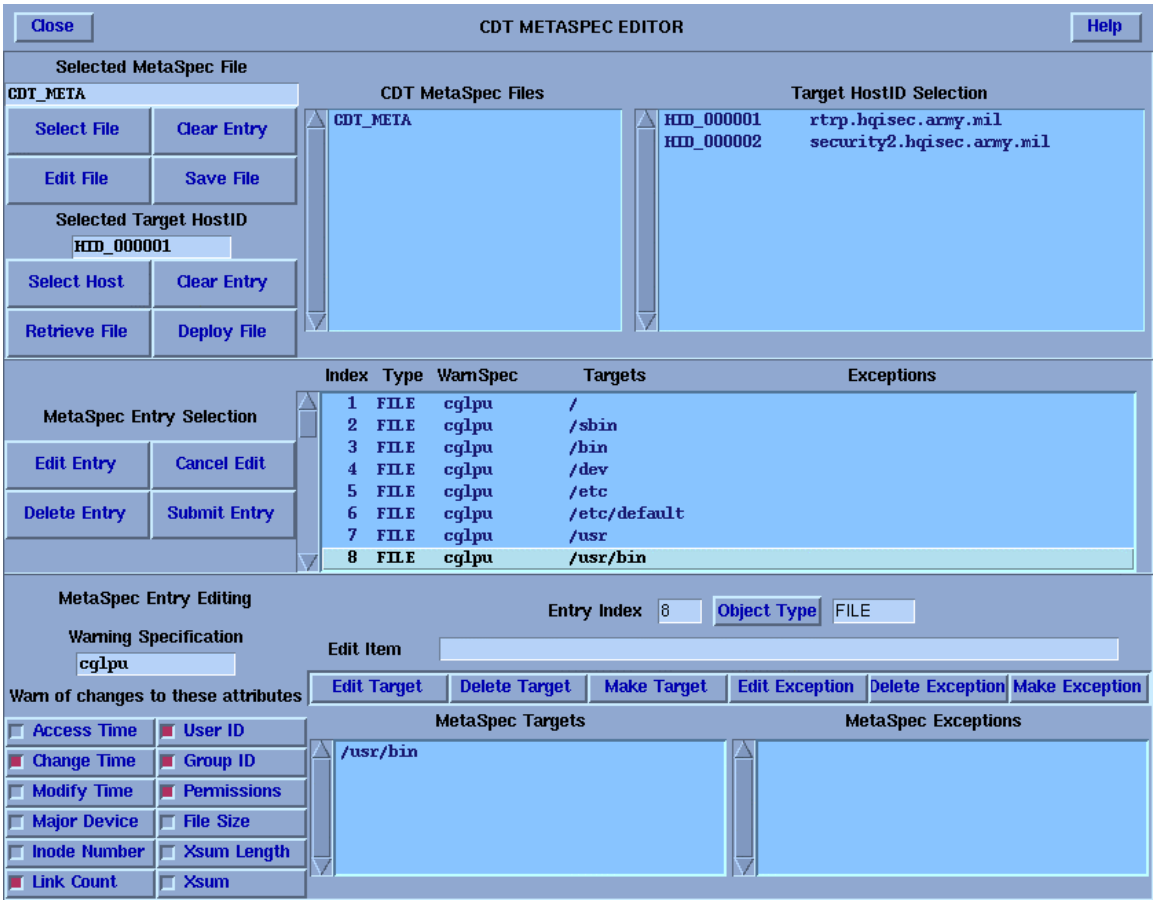


Figure 3-13. MetaSpec Editor.

Making changes to a CDT MetaSpec file are simple and straightforward if the proper procedures are used. The following steps are included to demonstrate how to make changes to the CDT MetaSpec File:

1) Selected a Target HostID.
2) Select a CDT MetaSpec File.
3) Press Edit File to apply MetaSpec file data to the MetaSpec Entry Selection area.
4) Select the MetaSpec line entry for editing by either double clicking the entry or selecting the entry and pressing the `Edit Entry` button. New MetaSpec entries can be created by either editing an existing entry or inputting the data directly into the MetaSpec Entry Editing area.
5) Edit the MetaSpec entry as desired. More details on this to follow.
6) Press `Submit Entry` to apply changes.
7) Press `Save File` to save changes to the MetaSpec File.
8) Press `Deploy File` to apply changed/new MetaSpec File to the system (command host or remote host).

Now that the general instructions have been discussed, the next item is to briefly discuss the MetaSpec File and the format of the entries. The MetaSpec File is essentially a text file consisting of a collection of MetaSpec line entries. Each entry has three required fields and an optional field as shown:

Type WarnSpec Target [Exceptions]

Type indicates what type of object is being examined. The three allowable types are File, User, and Group. A File type is used to identify file and directory items that are to be monitored for change. The User and Group types are used to identify those users, or groups, that are to be monitored.

WarnSpec (Warning Specification) defines the change detection attributes that are to be activated. For each type of metapec entry there are a unique set of change detection attributes.

Target indicates the file, user, or group that is to be examined for possible changes. Wildcards are available for all three MetaSpec types. Type, User, and Group can use the wildcard ALL to indicate all users or all groups. Type File can use standard UNIX wildcards to help define files and directories for monitorship.

Exceptions allow the user to omit particular files, users, or groups from the defined Target. Since wildcards are allowed in the Target specifications, the Exceptions area allows the operator the omit specific files from a wildcard.

Now that the basics have been explained lets more closely examine each of the MetaSpec types and associated warnspecs.

File Type. Figure 3-14 shows the MetaSpec Entry Selection and MetaSpec Entry Editing areas for MetaSpec entry of type FILE. For the type FILE, the allowable change detection attributes are:

(a) Access Time
(c) Change Time
(m) Modifiy Time
(d) Major Device
(i) Inode Number
(l) Link Count
(u) UserID
(g) GroupID
(p) Permissions
(s) File Size
(t) Xsum Length (Checksum Length)
(x) Xsum (Checksum)

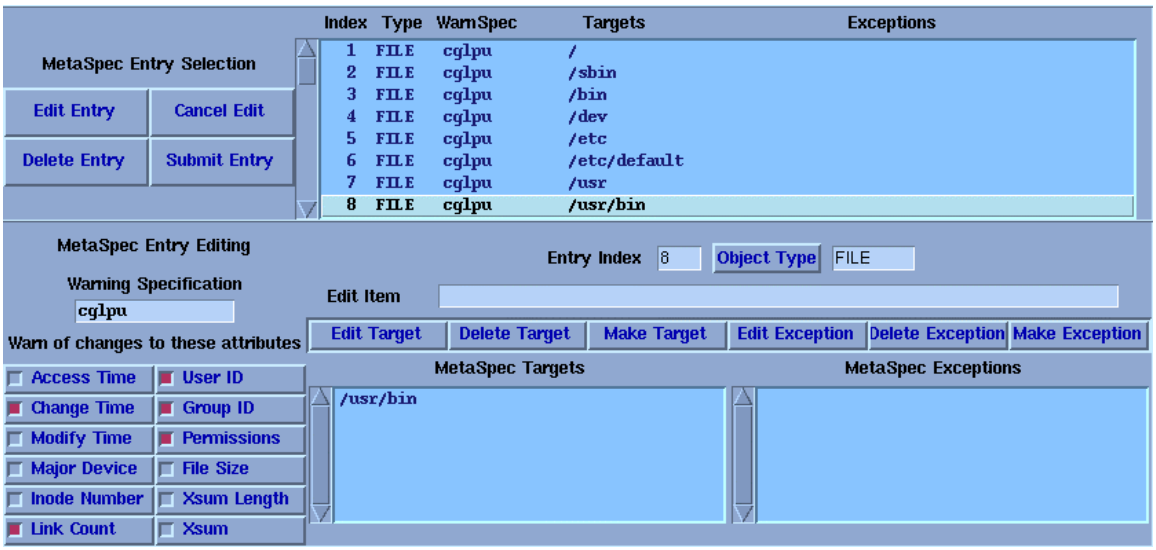


Figure 3-14. File MetaSpec Entry.

Targets for this file type are specified as either specific files or specific directories. However, if a directory is specified it does not include the files in that directory. To specify all the files is a directory use a standard UNIX wildcard such "/bin/*". The "*" wildcard can be used to specify any file(s) as desired.

User Type. The Warning Specifications for the User Type are much simpler than those associated with the File Type. Figure 3-15 shows the MetaSpec Entry areas for the MetaSpec type of User. For the type USER, the allowable change detection attributes are:

(u) UserID
(g) GroupID
(l) LoginDir
(s) Default Shell

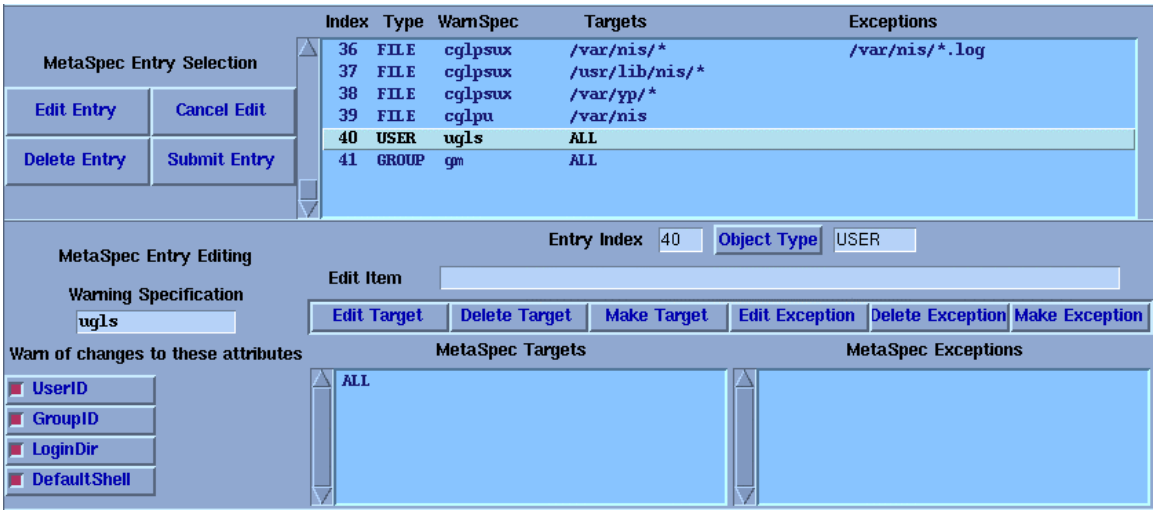


Figure 3-15. User MetaSpec Entry.

Targets for the User MetaSpec type are either specific user names, or the keyword ALL, which designates all users.

Group Type. The Warning Specifications for the Group Type are very simple. Figure 3-16 shows the MetaSpec Entry areas for the Group Type. For the type Group, the allowable change detection attributes are:

(g) GroupID (m) Members

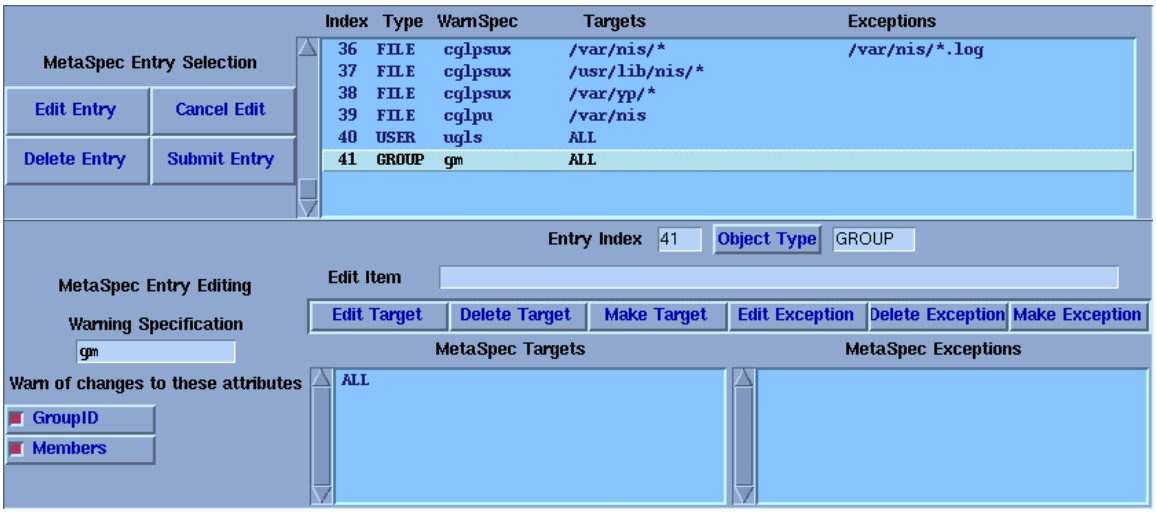


Figure 3-16. Group MetaSpec Entry.

Targets for the Group MetaSpec type are either specific group names or the keyword ALL, which designates all groups.

Now that is has been shown how the MetaSpec types and warning specifications are defined, lets spend a little time on the MetaSpec Entry Editing functions. Figure 3-17 shows the MetaSpec Entry Editing area. In this area the operator is allowed to specify the Entry Index (number), the Object Type (File, User Group), the Warning Specifications, and the MetaSpec Targets and Exceptions. The `Edit Item` field allows for the creation, modification, or deletion of MetaSpec Target and MetaSpec Exception information. This is accomplished by inputting the information into the field and using the `Edit Target, Delete Target, Make Target, Edit Exception, Delete Exception` or `Make Exception` buttons as appropriate.

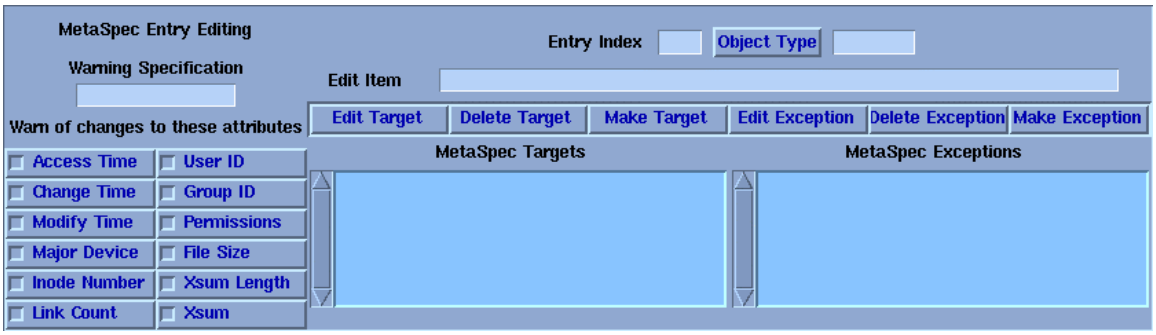


Figure 3-16. MetaSpec Entry Editing Area.

The following is a sample report that highlights the form and possible findings available using the CDT tool. Note that the report has been edited for space and contents.

```
************** SPI-NET 1.0 CHANGE DETECTION REPORT **************************

***** REPORT IDENTIFICATION SECTION *****

TARGET HOSTS REPORTING:

HostName         Start_TimeStamp    Final_TimeStamp    Elapsed_Time
--------         ---------------    ---------------    ------------
rtrp             19980406.135140    19980406.135146             06s

TARGET HOST SPECIFICATION:

HostName         OS_Type      OS_Name      OS_Version   Hardware
--------         -------      -------      ----------   --------
rtrp             UNIX         SunOS        5.5.1        i86pc

INSPECTION PARAMETER SPECIFICATION:

HostName         Inspector    Database Name        Metaspec File
--------         ---------    -------------        -------------
rtrp             CDT          spi_cdt_database     cdt_meta.spc

**** SECURITY FINDINGS SECTION *****

*** FINDING = ADDED ***

Host      Type  Item_Name        Activity
----      ----  ---------        --------
rtrp      FILE  /etc/hosts.equiv ADDED
rtrp      GROUP test             ADDED
rtrp      USER  user1            ADDED
rtrp      USER  user2            ADDED
rtrp      USER  user3            ADDED

*** FINDING = CHANGED ***

Host      Type  Item_Name        Attribute    New_Value       Old_Value
----      ----  ---------        ---------    ---------       ---------
```

```
rtrp      FILE  /.profile        CheckSum     3124C2D8551D29F 085A408D9FA7412
                                                E8DADFD465568   E5896B3BAE041
                                                88FE            5128
rtrp      FILE  /.profile        ChangeTime   19980406.135043 19970716.125918
rtrp      FILE  /.profile        FileSize     169             154
rtrp      FILE  /etc/group       InodeNumber  37876           37975
rtrp      FILE  /etc/inetd.conf  Permissions  -rwxr--r--      -rw-r--r--
rtrp      FILE  /etc/inetd.conf  ChangeTime   19980406.134821 19980224.122019
rtrp      FILE  /etc/rc0         GroupID      3               1
rtrp      FILE  /etc/rc0         ChangeTime   19980406.134454 19970702.143251
rtrp      FILE  /sbin/mount      OwnerID      0               2
rtrp      FILE  /sbin/mount      GroupID      103             2
rtrp      FILE  /sbin/mount      ChangeTime   19980406.134554 19970701.094731
rtrp      GROUP bin              MemberList   "root","bin","d "root","bin","d
                                                aemon","dreed   aemon"
                                                "
rtrp      GROUP scty             MemberList   "thendy","dreed
                                                ","jcrowley",
                                                "cruther"

SUMMARY OF FILE CHANGES:

HostName              Added  Deleted  Changed
--------              -----  -------  -------
rtrp                      1        0       20

SUMMARY OF USER CHANGES:

HostName              Added  Deleted  Changed
--------              -----  -------  -------
rtrp                      3        0        0

SUMMARY OF GROUP CHANGES:

HostName              Added  Deleted  Changed
--------              -----  -------  -------
rtrp                      1        0        2
```

3.5.4 Configuration Query Language (CQL).

The Configuration Query Language is not a true security tool like the others associated with SPI-NET. Instead, CQL is scripting language that allows the operator to create and implement system checks that are you unique to that system or an application on that system. As will be seen later on, the primary system testing routines used by the QSP are developed using CQL scripts. Similar routines and checks can be developed by the operator as required. The specifics on how to write, and develop these scripts, are outside of the scope of this document. For more information about CQL please refer to the SPI Reference Manual.

3.5.5 Password Security Inspector (PSI).

The PSI is a tool that examines the password file for weak and exploitable password entries. PSI attempts to crack passwords using variants of words from the user's password entry and by using various dictionary schemes.

Figure 3-18 shows the parameter screen for PSI. The lower right hand corner of the screen shows the parameter options for the Password Security Inspector.

Dictionary Level. PSI supports three dictionaries. These are the Local Dictionary, the Trivial Dictionary, and a Full Dictionary. These dictionaries can be found in the directory …/spin-1.01/binr/D/data/psi/.

Local Dictionary	Empty at install, the local dictionary can hold operator selected words, acronyms, etc.
Trivial Dictionary	The trivial dictionary includes common items not found in a normal or full dictionary. These items include names, insults, and popular characters.
Full Dictionary	The full dictionary is a simple English dictionary with approximately 30,000 words.

The Dictionary Level parameter is defined as follows:

Level 1 - Apply only Local Dictionary
Level 2 - Apply Local and Trivial Dictionaries
Level 3 - Apply Local, Trivial, and Full Dictionaries

Reverse Dictionary Words ? Selecting `Yes` causes PSI to additionally test the reverse of each dictionary word used in the crack attempt. Selection of this option will double execution time.

Up-Case Dictionary Words ? Selecting `Yes` causes PSI to also test the Uppercase of each dictionary word used in the crack attempt. Selection of this option will double execution time.

Display Found Passwords ? Selecting `Yes` allows the bad passwords to be printed in the output report. Selecting No causes the report to only print `<MATCHED>` instead of the actual password.

Use and Update a Logfile ? Employing a logfile avoids the retesting of passwords that are unchanged from the previous test. Using a logfile also enables the usage of password aging. If the use of a logfile is selected (`Yes`), a logfile name must be entered in the field `Password Logfile` Name.

Check Password Aging ? Selecting Yes causes PSI to check the age of the password against the password logfile. An age limit in days must be set in the field `Age Limit`.

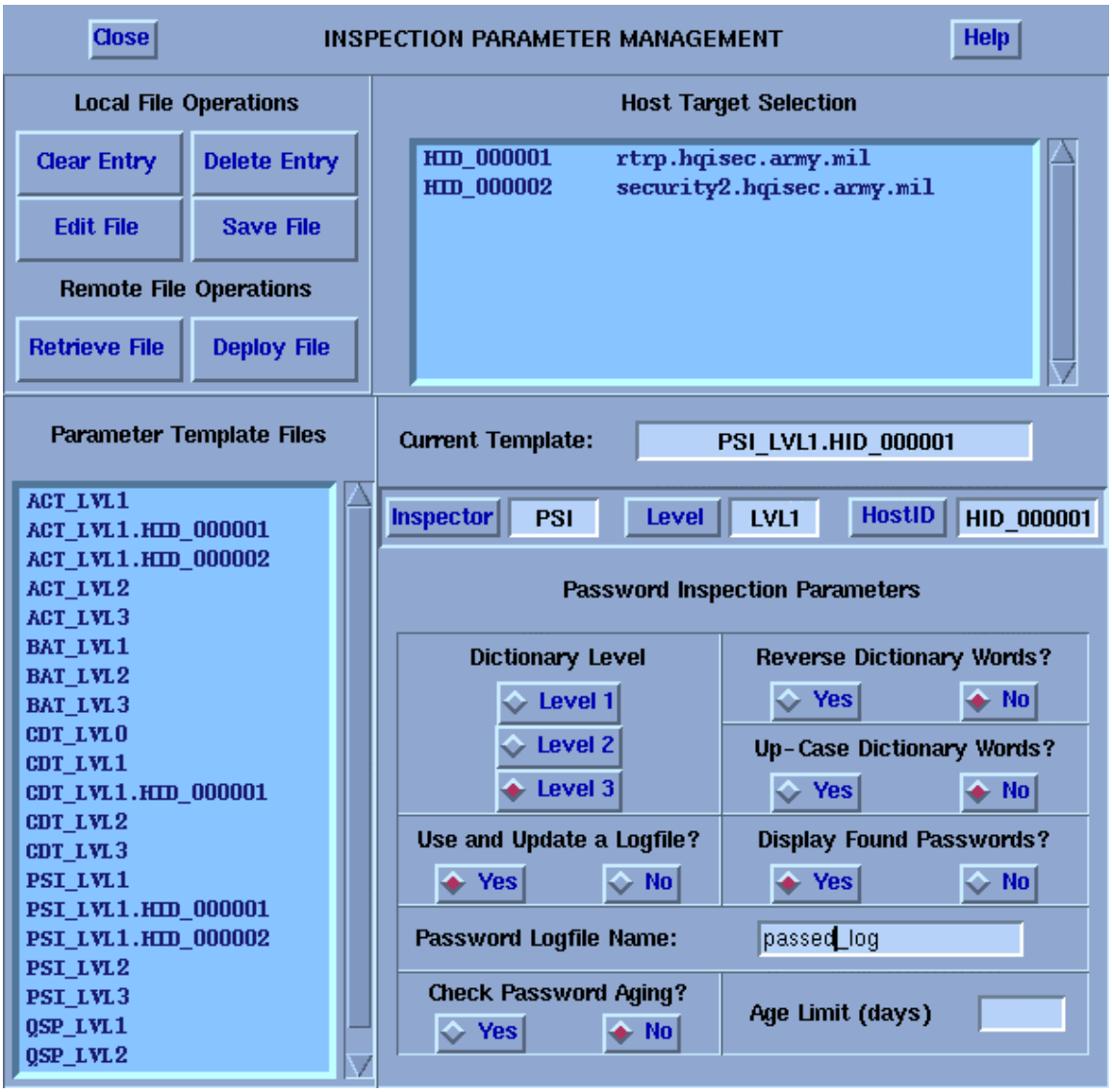


Figure 3-17. PSI Parameter Screen.

The following is a sample report that highlights the form and possible findings available from using the PSI tool. Note that the report has been edited for space and contents.

```
************ SPI-NET 1.0 PASSWORD INSPECTION REPORT ************************

===== REPORT IDENTIFICATION SECTION ===================================


TARGET HOSTS REPORTING:

HostName         Start_TimeStamp    Final_TimeStamp    Elapsed_Time
--------         ---------------    ---------------    ------------
rtrp             19980406.071435    19980406.071628         01m 53s


TARGET HOST SPECIFICATION:

HostName         OS_Type      OS_Name      OS_Version   Hardware
--------         -------      -------      ----------   --------
rtrp             UNIX         SunOS        5.5.1        i86pc


INSPECTION PARAMETER SPECIFICATION:

HostName         Inspector  User Database  Pswd Log File  Dictionaries
--------         ---------  -------------  -------------  ------------
rtrp             PSI        -NAV-          passed_log     psi_dict.full

===== SECURITY FINDINGS SECTION ======================================

*** FINDING = BAD_PASSWD ***

HostName        Subject Account_Name    Activity    Password
--------        ------- ------------    --------    --------
rtrp            USER    mreed           BAD_PASSWD  <NULL>
rtrp            USER    pdybvig         BAD_PASSWD  <NULL>
rtrp            USER    webuser         BAD_PASSWD  password

*** FINDING = NEW_ACCT ***

HostName        Subject Account_Name    Activity
--------        ------- ------------    --------
rtrp            USER    jcrowley        NEW_ACCT
rtrp            USER    pdybvig         NEW_ACCT
rtrp            USER    thendy          NEW_ACCT

*** FINDING = ROOT_PRIV ***

HostName        Subject Account_Name    Activity
--------        ------- ------------    --------
rtrp            USER    root            ROOT_PRIV
rtrp            USER    smtp            ROOT_PRIV

PASSWORD INSPECTION SUMMARY:

HostName         BadPasswords  NullPasswords   RootAccounts  AccountsTested
```

rtrp	3	2	2	21
GrandTotals	3	2	2	21

3.5.6 Quick System Profile (QSP).

Figure 3-19 shows the parameter screen for QSP. The QSP uses hundreds of specialized tests that check the operating system and file system for vulnerabilities. QSP contains the bulk of the tests performed by the program Computer and Oracle Password System (COPS) that are not covered by other SPI-NET tools. QSP is a CQL script that is located at /spin-1.01/binr/D/cqls/cql_qsp_ck. Those with sufficient experience with UNIX and CQL scripts can adjust QSP to better suit the needs of the system. Some of the standard QSP tests include:

a. Permissions - files and directories
b. cron files - pathnames and writability
c. /etc/rc* - pathnames and writability
d. Anonymous FTP - home directory not root
e. ftpusers - all root equivalent accounts
f. Users - home directories, dormant users, configuration files (.rhosts, .profile, etc)
g. Groups - duplicated names, GIDs
h. Passwords - fields, duplicates, null

Include Check for Group Access ? The standard QSP performs many checks for "world" access permissions but does not check for "group" access permissions. Selecting Yes causes QSP to include "group" checks along with "world" access permission checks.

Scan Entire Disk for Vulnerabilities ? Standard QSP only checks selected system files and services. This option checks for other vulnerabilities that may exist outside these standard checks. These additional checks include:

1) Files owned by unknown users or groups.
2) Files with suspicious names.
3) World writable Set-UID binaries.
4) All Set-UID shell scripts.
5) Any "device" files not located in the systems /dev directory.

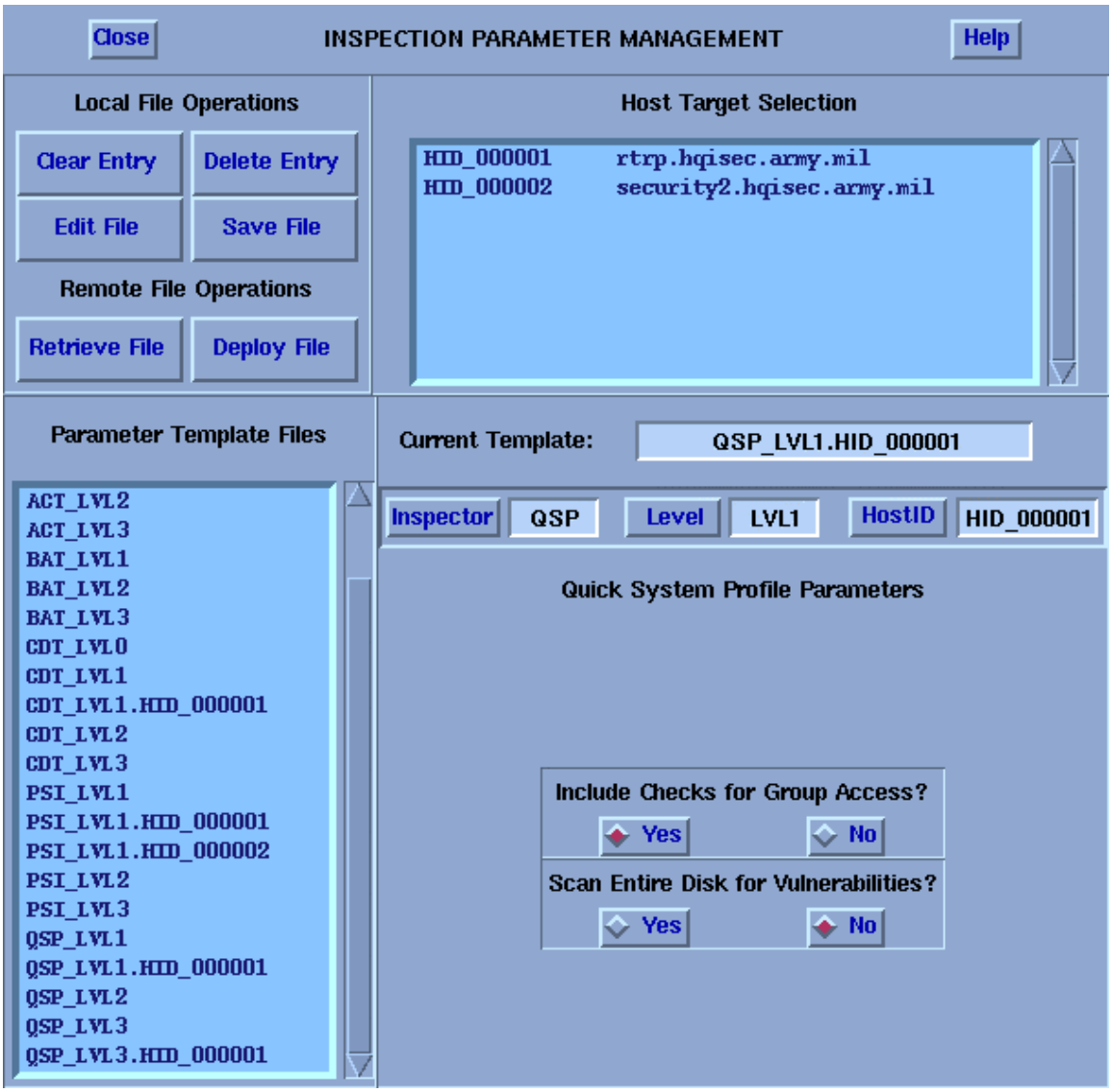


Figure 3-18. QSP Parameter Screen.

The following is a sample report that highlights the form and sample findings available using the QSP tool. Note that the report has been edited for space and contents.

```
************** SPI-NET 1.0 SYSTEM PROFILE REPORT ****************************

===== REPORT IDENTIFICATION SECTION ========================================

TARGET HOSTS REPORTING:

HostName         Start_TimeStamp    Final_TimeStamp    Elapsed_Time
--------         ---------------    ---------------    ------------
rtrp             19980406.090555    19980406.090602             07s


TARGET HOST SPECIFICATION:

HostName         OS_Type      OS_Name      OS_Version   Hardware
--------         -------      -------      ----------   --------
rtrp             UNIX         SunOS        5.5.1        i86pc


INSPECTION PARAMETER SPECIFICATION:

HostName         Inspector  CQL Script     Parameters
--------         ---------  ----------     ----------
rtrp             CQL        cql_qsp_ck     on

===== SECURITY FINDINGS SECTION ============================================

*** Bad PERM Value -- Vulnerability ***

    The following important files or devices were found to be either
    writable or readable by group or world, as indicated.  In some of
    these cases, the file was flagged because it was referenced within
    a root auto-exec or boot-up script (e.g., /etc/rc.boot).  These
    scripts should be examined to assess their dependency upon the
    indicated file.

HostName       Name             Bad Value      Comment
--------       ----             ---------      -------
rtrp           /dev             "drwxrwxr-x"   should not be group writable
rtrp           /etc             "drwxrwxr-x"   should not be group writable
rtrp           /etc/default     "drwxrwxr-x"   should not be group writable
rtrp           /var/mail        "drwxrwxrwt"   file listed in rc boot file ( /
                                                  etc/rc0.d/K57sendmail ) is wo
                                                  rld writable

*** Request ID=ftp_service ***

    If a system provides an anonymous FTP service, the ftp directory
    should be owned by root, and its subdirectories (bin, pub, etc,)
    should not be world writable.  In addition, there should be no
    root or root-priviledged ftp accounts.  A system file called
    "ftp.deny" should contain the names of all accounts which are
    not to be ftp accounts.  (Perversely, this file is actually named
    "ftpusers" on some systems.)
```

```
HostName       Name             Problem    Comment
--------       ----             -------    -------
rtrp           /etc/ftpusers    NOEXIST    ftp users file /etc/
                                             ftpusers should ex
                                             ist

*** Request ID=cmd_shell ***

    System accounts (non-login accounts) should not have an empty shell
    field.  Set empty shell fields in the password file to "/bin/false".

HostName       Account      Problem    Comment
--------       -------      -------    -------
rtrp           lp           SHELL      Empty shell field.  Use /bin/false
rtrp           nobody       SHELL      Empty shell field.  Use /bin/false

*** Eval ID=groupdb_test ***

    Few installations employ group account logins, but it is still a
    good precaution to "star-out" (*) the group password fields.

HostName       File         Problem         Comment
--------       ----         -------         -------
rtrp           /etc/group   check_grouptab  Group scty has no password
rtrp           /etc/group   check_grouptab  Group staff has no password
rtrp           /etc/group   check_grouptab  Group sys has no password

*** Eval ID=userdb_test ***

    This section will report user accounts that are either ill-formed
    or otherwise represent poor security practice.  A common warning
    involves multiple user accounts assigned the same user-id.  One
    should ensure that individual users are assigned unique user-ids.
    (Exception: root users (uid=0), but these should be minimized.)

HostName       File         Problem         Comment
--------       ----         -------         -------
rtrp           /etc/passwd  check_usertab   User smtp has uid=0 and is not
                                               root
*** PRIVILEGED USERS ***

    The following user accounts are root-equivalent.

HostName       UserName        Comment
--------       --------        -------
rtrp           "root"          This user has a root account
rtrp           "smtp"          This user has a root account

--- CQL PROCESSING ERRORS -----------------------------------
```

3.6 Remote Host Configuration & Security Domains.

In this section we will discuss how to create, and add hosts, to the security domains.

SPI-NET can use security domains to control and manage the SPI-NET tools on the command host, and on affiliated remote hosts. In a standalone implementation, a security domain of just the command host is automatically created. To add a remote host to a SPI-NET security domain, five additional tasks must be accomplished. These are:

1) The remote host must know the hostname of the command host. This is usually done as part of the installation process.
This information is located in file /spin-1.01/binr/D/HOSTINFO/Host_Table. The command host is always identified as HID_00001.

2) A HostID must be assigned to the remote host using `the Domains -> Assign HostIDs`. The command host is always assigned HostID HID_000001. Figure 3-20 shows the Host ID Assignment window. To assign a HostID, enter the remote host information (HostID and Hostname) in the `Current Entry` area. HostIDs have the format HID_###### and each host in the security domain must have a unique HostID. Use the `Submit Entry` button to apply the new entry to the SPI-NET Host Table. To save the changes to disk, select the `Commit Table` button. Changes to the SPI-NET Host Table can be confirmed by selecting `the Reread Table` button that rereads the SPI-NET Host Table from disk.

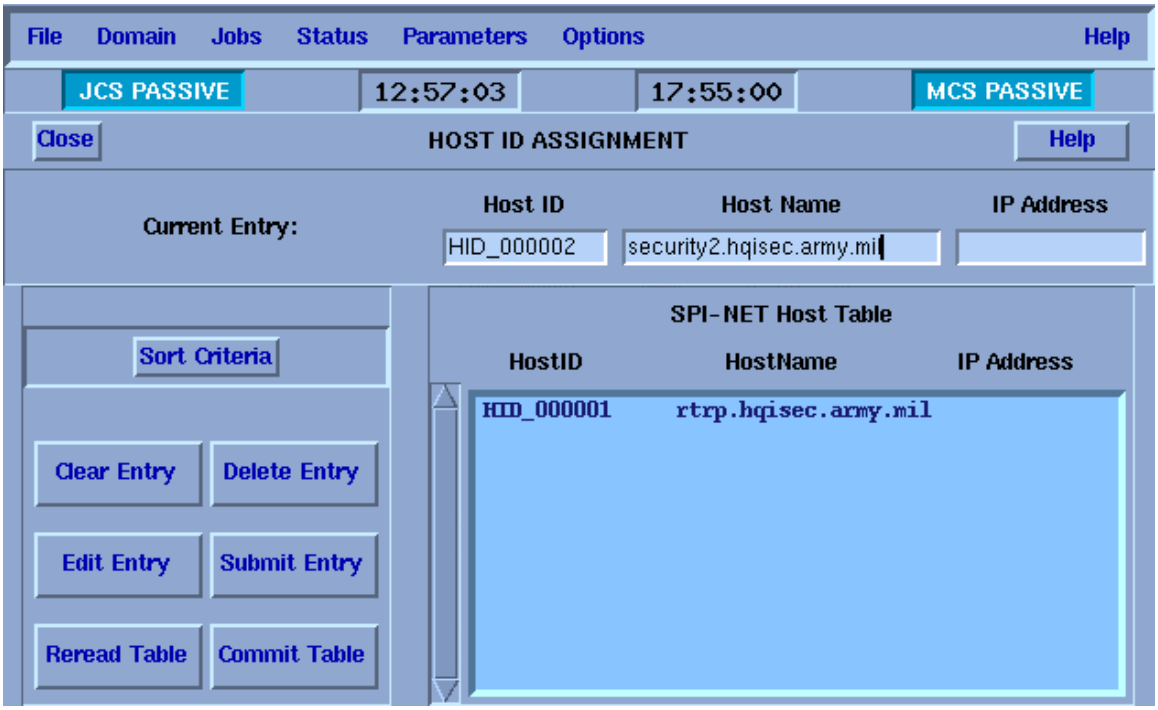


Figure 3-19. Assigning HostIDs.

3) Generate a Digital Signature Standard (DSS) certificate for the remote host using `Domain -> DSS` Certificates. Figure 3-21 shows the SPI-NET Certificate Manager that is used to generate and expire DSS certificates. DSS certificates, for remote hosts, are generated by first selecting the HostID (double click) so that the remote host appears in the `Target HostIDs` area. Second, select the remote HostID from the `Target HostIDs` area. Third, press the

`Generate/Assign Certificates` button to create new certificates for the remote host. With the DSS certificate generated, new entries for the remote host should appear in the `Current Certificate Listings` area. If needed, existing certificates for a specific host can be EXPIRED using the `Expire Assigned Certificate` button.

4) With DSS certificates for the remote host generated, specific public and private keys must be transferred from the command host to the remote host. Specifically, *move* the remote hosts private key (crt_######.prv) from the command host to the remote host where ###### is the HostID number of the remote host. Also *copy* the public key (crt_000001.pub) from the command host to the remote host. The keys should be taken from the command host, from the following directory …/spin-1.01/*binm*/D/CERTINFO/ and placed in the remote host, from the following directory …/spin-1.01/*binr*/D/CERTINFO/.

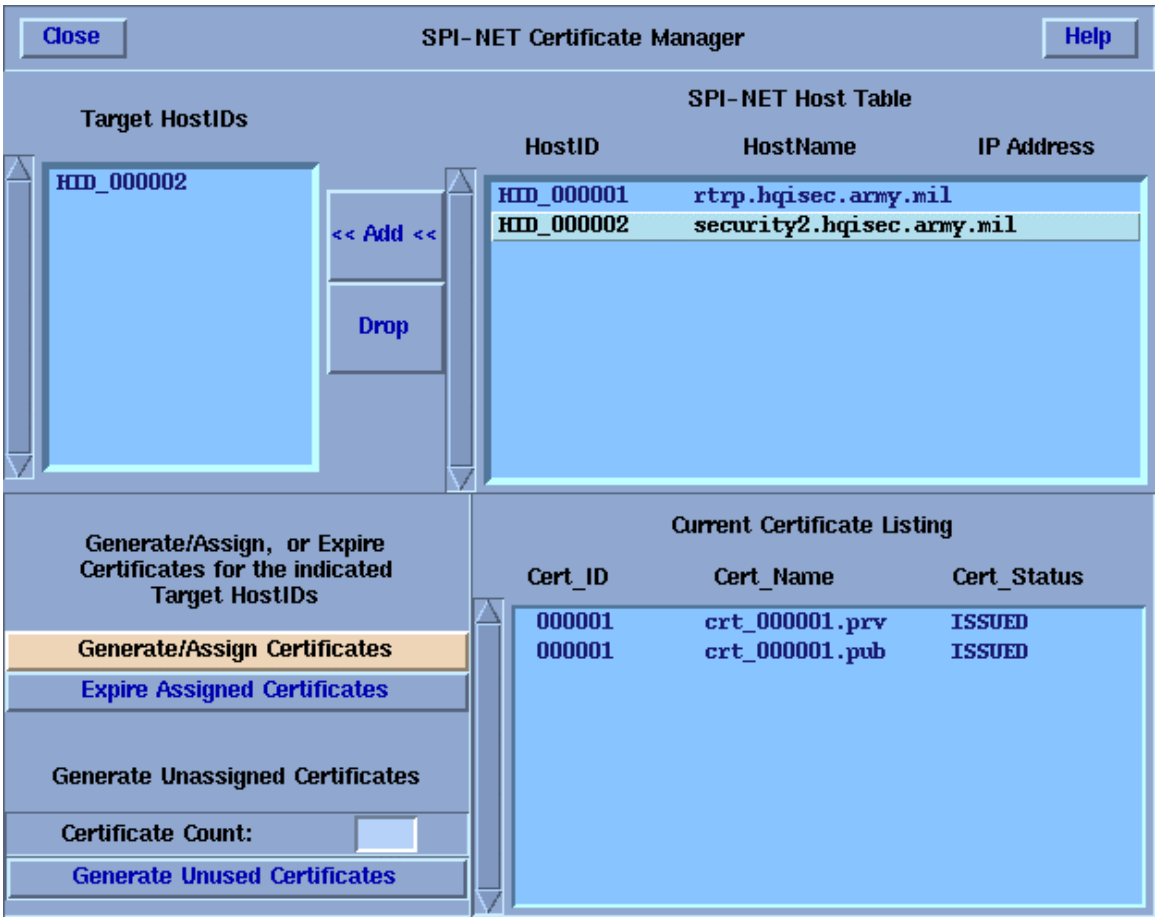


Figure 3-20. DSS Key Generation.

5) Lastly, with the DSS certificates generated, and the keys transferred to the correct locations, it is time to add the remote host to a HostGroup or HostGroups. Hosts are assigned to Host Groups using `Domain -> Define HostGroups` as show in Figure 3-22. To add a remote host to the HostGroup, (1) select the HostGroup (`Defined Host Group` area) to which the remote hosts is to be added or enter a new HostGroup in the `Selected HostGroup` field; (2) select a

host from the SPI-NET Host Table; (3) press `<< Add <<` to add the remote host to the selected HostGroup; (4) Press `<< Accept` to accept changes to the HostGroup entry. To remove a host from a HostGroup, select the HostGroup, select the host to be removed from the `HostGroup Members` list, press `>> Delete >>,` and finally press `<< Accept` to accept the deletion.

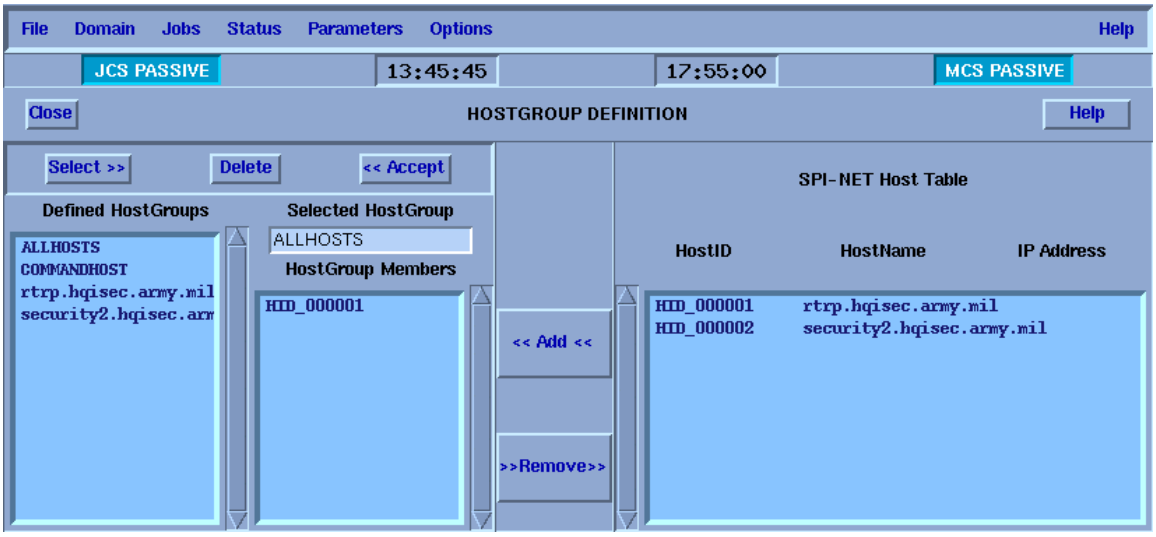


Figure 3-21. HostGroup Assignment.

3.7 Recommendations. Since SPI-NET has a very good built-in scheduling feature, the SPI-NET security tools can be run at anytime without operator intervention. The following are the defaults that are created upon installation for the running of the SPI-NET tools:

CDT	Every 12 Hours
ACT	Every 12 Hours
PSI	Every Day
BAT	Every 7 Days
QSP	Every 7 Days

These are good starting points for the tools. However, usage of the tools must be adapted to suit the system on which they are being run. Additionally, it is of little good to have the tools running if the reports are not reviewed by an operator or System Administrator.

4.0 Transmission Control Protocol Wrapper.

TCP Wrapper is a commonly used, publicly available, security tool that creates a host-based firewall for your system. Through the usage of TCP Wrapper, the user can control and log which hosts have access to which services on your system. With this package System Administrators can monitor and filter requests for systat, finger, FTP, Telnet, rlogin, rsh, exec, TFTP, talk and other network services. TCP Wrapper also provides a logging capability to many network services that are not normally provided by the system.

4.1 Basic Features.

TCP Wrapper is a program used to help shield systems from suspicious and malicious network intrusions. The TCP Wrapper has no impact on authorized users and does not require any changes to existing systems software code. Only minor changes to several system configuration files are required. TCP Wrapper runs in the background and has a minuscule impact on system performance. The TCP Wrapper program does not exchange any data with the network client or server processes, making the process more secure. TCP Wrapper can be configured to send its data to the syslog for logging, and can be used in conjunction with the swatch utility to provide active monitoring of network access to the system in real-time.

4.2 Operating Basics.

In simple terms, TCP Wrapper acts as a gate guard checking all incoming connection requests against the access roster. All connection requests are logged whether they are allowed or denied. Those connection requests not on the access list are refused entry while those on the access list are escorted to their intended destination/service. Technically, through changes in the inetd.conf file, TCP Wrapper is logically placed between an incoming connection request and the actual service being accessed. When a service connection is requested, TCP Wrapper is started and the following actions occur:

1) TCP Wrapper compares the incoming Internet Protocol (IP) address and requested service with an access list to see if this host/service combination should be allowed or denied. If it is denied, TCP Wrapper will drop the connection.

2) TCP Wrapper can optionally send a “banner” to the connecting host. Banner message sent can vary depending on whether access was allowed or denied.

3) The results of the connect action are logged using the syslog service.

4) Any “denied” connections are now dropped. For connections that are to be allowed, TCP Wrapper calls the “*real*” network daemon using the same initial parameters presented to TCP Wrapper. At this point, the real network daemon begins activity and TCP Wrapper is no longer involved.

Access control is accomplished through the usage of two special files, *hosts.allow* and *hosts.deny*. These files contain the access lists for the systems. Each file consists of a series of entries detailing which host and service combinations are allowed or denied access.
In addition to the basic TCP Wrapper application, two additional utilities are included in the distribution that aid the System Administrator in managing the TCP Wrapper application. These utilities are *tcpdchk* and *tcpdmatch*. The tcpdchk utility will scan through your inetd.conf and TCP Wrapper configuration files for common problems and errors. The tcpdmatch utility allows the System Administrator to test hosts/service pairings against the access control logic to determine whether the host/service combination will be allowed or denied.

4.3 Installation.
TCP Wrapper is the most difficult and intrusive of the three mandated UNIX C2 Protect Tools to install. In order to properly install TCP Wrapper, changes are required to the Makefile to tailor the executables to the system. In addition to the tailoring of the Makefile, modifications are required to the syslog and inetd configuration files. Even though pre-compiled binaries of TCP Wrapper may be available from various sources, it is not advised that these be used. First, pre-compiled binaries are not tuned to your system and do not have all of the recommended modifications. Secondly, TCP Wrapper is distributed in an uncompiled or source code format. Any pre-compiled distribution was made by an unknown entity, with unknown trust.

The following steps detail a process that will perform a complete installation of TCP Wrapper which include unpacking the source code, modifying the Makefile, building the executables, and editing the syslog and inetd configuration files. The additional steps required to start and implement TCP Wrapper are covered in Section 4.4 Implementation.

1) Download the source code file (tcp_wrappers_7.6.tar.gz) from the ASSIST's FTP site or other suitable location. The most current version of TCP Wrapper is version 7.6. Since TCP Wrapper will create its own sub-directory as part of the installation process, place the file in a directory under which you will want this tool to be created. For the purposes of this guide the directory /secure will be used as an example.

NOTE: The sub-directory created is not permanent and can be removed after successful installation if desired.

```
http://www.assist.mil                          OR
ftp://ftp.assist.mil/pub/tools/tcp_wrappers/
```

2) Uncompress the TCP Wrapper file using the appropriate command:

```
uncompress tcp_wrappers_7.6.tar.Z              OR
gunzip tcp_wrappers_7.6.tar.gz
```

3) Extract the source code from the tcp_wrappers archive file (tcp_wrappers_7.6.tar) Note that a tcp_wrappers subdirectory will be created as part of the extract process.

tar -tvf tcp_wrappers_7.6.tar	View archive and check for archive integrity
tar -xvf tcp_wrappers_7.6.tar	Extract contents of the archive file

Change directory to tcp_wrappers_7.6

```
cd tcp_wrappers_7.6
```

4) Read the README file. The README file explains how TCP Wrapper works, where the logging information goes, and options that can be complied into the executables.

5) Copy the original Makefile to another file name, and keep the original copy.

```
cp Makefile Makefile.orig
```

6) Do a change mode on Makefile. This will allow editing of the Makefile.

```
chmod 744 Makefile
```

7) Modify the Makefile using the editor of your choice.

```
vi Makefile
```

There are two installation sections in the Makefile. These are the Advanced installation area and the Easy installation area. For our purposes, we will use the Advanced installation procedures. The Advanced installation option calls for the vendor-provided daemons to be left alone, and the inetd.conf file to be edited. In doing an Advanced installation, the TCP Wrapper must know the default directory of where the real network daemons are located so that they can be called and execute upon a successful connection to the system. The default network daemon directory is specified by the REAL_DAEMON_DIR.

Uncomment the appropriate REAL_DAEMON_DIR line for your operating system. Specification of the REAL_DAEMON_DIR is mandatory. If the listed entries do not match your system then an entry can be edited or added to reflect the location of you system network daemons.

```
# Ultrix 4.x SunOS 4.x ConvexOS 10.x Dynix/ptx
#REAL_DAEMON_DIR=/usr/etc
#
# SysV.4 Solaris 2.x OSF AIX
#REAL_DAEMON_DIR=/usr/sbin
#
# BSD 4.4
```

```
 #REAL_DAEMON_DIR=/usr/libexec
 #
# HP-UX SCO Unicos
#REAL_DAEMON_DIR=/etc
```

Next, scroll through the operating system templates area. Find and note the operating system identifier that correlates to your operating system. The operating system identifiers are located on a line by themselves followed by a ":". Note the exact spelling of the identifier as this information will be required later. If your system is not listed (or something does not come close enough), you have to edit the system dependencies section and do a make other rather than a make system_type

The System Dependencies sections are as follows:

```
TLI (transport-level interface) support
Differences between ranlib (1) and ar (1) implementations
Routines that are not present in the system libraries
Selection of non-default object libraries
System-specific compiler flags
Working around system bugs
Whether or not your system has NIS (or YP) support
Whether or not your system has vsyslog()
```

Read the text in the Makefile and select the appropriate options for your operating system.

If your operating system is listed, continue to scroll down the file, to the section on language extensions.

Uncomment the following line to enable the use of Banners.

```
STYLE= -DPROCESS_OPTIONS
```

Continue to scroll down the file until you get to the line that addresses the logfiles and the syslog facility.

Edit file and make changes to the FACILITY and SEVERITY LEVEL as represented below. Normally the entry for the syslog facility is FACILITY=LOG_MAIL which would place all the TCP Wrapper in the same log area as the mail log items. LOCAL0 through LOCAL7 are user definable system facilities that allow the System Administrator to label and thus segregate the unique syslog data into separate log files. If the facility LOCAL0 is already being used by the system, choose another LOCAL# facility. If you do not utilize LOCAL0, note the change and make the corresponding changes when the syslog.conf file is edited.

```
#
FACILITY = LOG_LOCAL0          #Normally FACILITY=LOG_MAIL
#
SEVERITY LEVEL=LOG_INFO        #Should be the default
```

```
#
```

Continue to scroll down the file to the section dealing with access control table pathnames:

```
TABLES= -DHOSTS_DENY=\"/etc/hosts.deny\" -DHOSTS_ALLOW=
\”/etc/hosts.allow\"
```

Edit the above entry to change the default placement of the hosts.allow and hosts.deny files. By default, the Makefile creates an executable that looks for these files in the /etc directory. This directory is very large and these files can get lost in the congestion. Making another directory in /etc will allow for ease of administration.

Suggestion: Use "/etc/wrap/hosts.deny" and "/etc/wrap/hosts.allow”.

```
TABLES= -DHOSTS_DENY=\"/etc/wrap/hosts.deny\" -DHOSTS_ALLOW=
\”/etc/wrap/hosts.allow\"
```

NOTE: Remember where you are telling TCP Wrapper to look for these files. This information will be required later in installation process and as part of the configuration process.

Scroll to the next section that addresses name/address conflicts.

This section asks TCP Wrapper to do a Domain Name Server (DNS) check of the hostname if the PARANOID option is enabled. With this option on, connections will be rejected when the hostname does not match the host. This option tries to protect against hosts that claim to have someone elses hostname.

By default, this option is enabled. However, if there are any doubts, disable this option. To disable this option, comment out the line as indicated below.

```
#PARANOID= -DPARANOID
```

Scroll down the file to the following line and uncomment the following line:

```
KILL_OPT= -DKILL_IP_OPTIONS
```

NOTE: This option is not needed for modern UNIX systems that can stop source-routed traffic in the kernel. However, even if the kernel is performing this task, the usage of this option will not adversely impact the system.

Write the changes to the file and quit the editor. At this point, TCP Wrapper is ready to be compiled using the above options made to the Makefile.

8) To compile the TCP Wrapper executables, type the following at the command prompt:

make *system_type* Example: make linux

system_type correlates to the operating system identifier noted previously.

9) The next step in the configuration process is to move the executables (tcpd, tcpdchk, tcpdmatch) to /etc/wrap/ or other appropriate directory. Additional copies of tcpdchk and tcpdmatch utilities can be moved to the directory area where the hosts.deny and hosts.allow files are built and tested. Moving these files, to this additional location, allows ease of maintenance and administration.

```
cp /secure/tcp_wrappers_7.6/tcpd /etc/wrap/
cp /secure/tcp_wrappers_7.6/tcpdchk /etc/wrap/
cp /secure/tcp_wrappers_7.6/tcpdmatch /etc/wrap/
```

At this point, the executables have been complied with the options that you have chosen and now you are ready to configure the system to work with TCP Wrapper.

10) The installation process is not complete until the system is configured to use TCP Wrapper. The first step in this configure action is to make copies your original inetd.conf file and place them in inetd.conf.old and inetd.conf.new. This will preserve your original configuration (inetd.conf.old) so that it can be restored if the new version (inetd.conf.new) gets corrupted, destroyed, or does not work. Copy the inetd.conf file using the syntax below:

Change Directories to the location of your inetd.conf file:

```
cd /etc
```

Make copies of the original inetd.conf file:

```
cp inetd.conf inetd.conf.old
cp inetd.conf inetd.conf.new
```

Perform chmod and modify the inetd.conf.new file using your editor of choice:

```
chmod 644 inetd.conf.new inetd.conf
vi inetd.conf.new
```

Since changes are being made to the inetd.conf file, this is a good time to review the configuration file to determine if all of the currently active services are necessary or required by the system.

Scroll down to the line containing the first service. (e.g., FTP)

Decide if this service will be used. If this service is not one that will be used or needed, place a pound sign (#) in front of that line. Scroll to the next line and evaluate.

When you reach a line with a service that you will be using, perform the following steps:

? Insure that the line does not have a pound sign (#) in front of the line.

? Move your cursor to the 6th field and replace the contents (daemon) with the location (full pathname) of the TCP Wrapper executable. If the original 6th field indicates a directory path that is not the default daemon directory then the full daemon path must be used in the 7th field. Additionally, any daemon options originally presented in the 7th field should be retained by the 7th field.

Example:

Before Changes

```
ftp stream tcp nowait root /usr/sbin/in.ftpd in.ftpd
```

After Changes

```
ftp stream tcp nowait root /etc/wrap/tcpd    in.ftpd
                           ^^^^^^^^^^^^^^^^
```

NOTE: /etc/wrap/ was the sub-directory that was chosen to place the TCP Wrapper executables.

As noted previously, to override the default directory location of the system daemons (7th field), as specified in the TCP Wrapper Makefile, use full pathnames, such as /usr/sbin/in.ftpd.

Continue this process for each service listed in the inetd.conf.new file. It should be noted that the usefulness of TCP Wrapper for User Datagram Protocol (UDP) based services is limited. As opposed to TCP based services, UDP services are not connection oriented and once the service is opened it will remain open for a period of time. As a result, an original request for a UDP service will be evaluated by TCP Wrapper and a log entry made. However, since UDP services do not immediately close, any subsequent requests for that service, prior to closure, may not properly be evaluated and logged by TCP Wrapper.

After you have completed this process, write your changes to the file and exit.

11) The next step in the configuration process is to add an entry to the syslog configuration file to allow proper placement of TCP Wrapper log output. First, make copies of our original syslog.conf file:

Change the directory to the location of your syslog.conf file:

```
cd /etc
```

Make copies of the syslog.conf file:

```
cp syslog.conf syslog.conf.old
cp syslog.conf syslog.conf.new
```

Perform chmod and edit the syslog.conf.new file accordingly:

```
chmod 644 syslog.conf.new syslog.conf
vi syslog.conf.new
```

Add the following line. (Place this line in a manner that global entries (e.g. *.info) do not take precedence.)

```
LOCAL0.info          /var/log/tcpd.log
```

This line defines the facility, severity level, and location of the TCP Wrapper logfile. LOCAL0 refers back to the change in the TCP Wrapper Makefile. The line entry should be modified to represent a proper location for the logfile based on your system configuration. Some systems may want to place the file in the /var/adm or /var/adm/syslog area.

Save the modifications and exit the file.

4.4 Implementation.

4.4.1 Initial Operational Capability.

The following steps outline the procedures needed to bring TCP Wrapper to an initial operational capability. Initial operation only starts the TCP Wrapper process and begins the accumulation of connection data in a TCP Wrapper log file. Initial operation does not include any access control logic or filtering. Implementation of the access control logic, and associated configuration files, is found in Section 4.4.2 TCP Wrapper Operating Concepts and Section 4.4.3 Building Initial Access Control Files.

1) A logfile must be created so that syslog can begin writing to the TCP Wrapper log file. Use the name and directory location that were specified by the new TCP Wrapper entry inserted into the syslog.conf.new file. To create the logfile and set permissions for the tcpd.log, perform the following commands:

```
touch /var/log/tcpd.log
```

```
chmod 644 tcpd.log
```

Be sure to use the full path name of the file, or perform the touch command in the directory you wish the tcpd.log to reside. After you have performed the touch command, creating the tcpd.log file, ensure that the file exists and has the correct permissions.

2) Prior to initially starting TCP Wrapper, the inetd.conf and syslog.conf must be replaced with the new versions and chmod performed to return the file permission settings back to the original settings. For future purposes remember that the .conf.old version is the original version, .conf.new version is the edited version, while the .conf is the current version to be used by the system.

```
cd /etc
cp inetd.conf.new inetd.conf
cp syslog.conf.new syslog.conf
chmod 400 syslog.conf syslog.conf.new
chmod 400 inetd.conf inetd.conf.new
```

3) To enable the new inetd.conf and syslog.conf files, find the process identifier (PID) and restart both the syslogd and inetd daemons using the following commands:

```
ps -aux     grep syslogd     : note syslogd pid
ps -aux     grep inetd       :  note inetd pid
```

The PID for each of these processes will be displayed to the screen.

```
kill -HUP "pid#"          Example: kill -HUP 156
```

This command tells the syslogd or inetd daemon to reread the configuration file so that the changes that were made will take effect.

4) Run the tcpdchk utility to examine the inetd.conf file for errors and check the TCP Wrapper setup. The tcpdchk utility can be run at any time to check the TCP Wrapper and inetd configuration.

```
cd /etc/wrap/
./tcpdchk
```

5) TCP Wrapper should now be operating at a very basic level. With no configuration files in effect, TCP Wrapper will accept any incoming network connection as specified by the inetd.conf. However, TCP Wrapper will log all incoming connections in the logfile specified in the syslog.conf.

6) Telnet back into the machine from a remote location to test the basic implementation of TCP Wrapper. After you have entered the system, check the tcpd.log logfile for data. If log entries exist then TCP Wrapper was properly installed. In no entries exist, review the installation process to check

to that the installation procedures where properly followed and ensure that no misspelled or incorrect data was used in the installation process.

Before building the configuration files, hosts.allow, or a hosts.deny file it is a good idea to let TCP Wrapper run for 3 to10 days. This will serve two purposes :

1. It will allow you to see what machines connect to your host and what services are being utilized. This will give you an idea of what services and IP address to place in your hosts.allow file after the initial test period is over. You can access this information by viewing your tcpd.log file.

2. This period also allows you to make sure that TCP Wrapper was installed correctly and that all the logfiles are being written to, and all the daemons can find the files that they must access.

4.4.2 TCP Wrapper Operating Concepts.

Prior to actually writing the hosts.allow and hosts.deny, there are several syntax and procedural items that need to be addressed. First, all lines in the hosts.allow and hosts.deny are processed in order of appearance. Only the first match between the incoming connection request and a line entry is used. Blank lines or lines beginning with a "#" are ignored. When an incoming connection is handled by TCP Wrapper, the program applies the following basic rules:

1) The hosts.allow file is searched to see if this host/service pair should be allowed.
2) If no match is found, the hosts.deny file is searched to see if the host/service pair should be denied.
3) If no match is found in either the hosts.allow or the hosts.deny files then the connection is allowed.

The general format used to create the line entries in the hosts.allow and hosts.deny is :

```
daemon_list : client_list [: option : option]
```

daemon_list	list of services applicable to this entry
client_list	list of clients applicable to the entry

NOTE: Elements in the daemon list and the client list should be separated by blanks or commas.

Example #1 - Simple Telnet and FTP:

```
# /etc/wrap/hosts.allow
#
in.telnetd, in.ftpd : 137.99.90.1, 137.109.40.2
#
```

For this example, four specific conditions will be matched. If any of these four conditions are met, access will be allowed. The four conditions are:

(1) 137.99.90.1 Telnet access
(2) 137.99.90.1 FTP access
(3) 137.109.40.2 Telnet access
(4) 137.109.40.2 FTP access

If this line entry was located in a hosts.deny file then these four conditions would be rejected rather than accepted.

Specifying each and every host and service combination required by the hosts.allow and hosts.deny files would be a long and dreary task. However, TCP Wrapper does allow the usage of domains, IP subnets, and wildcards to ease the chore of configuration. TCP Wrapper allows the use of three different wildcard keywords as part of the hosts.allow and hosts.deny configuration.

ALL	The universal wildcard that matches anything and everything.
LOCAL	Matches any host whose name does not contain a ".".
EXCEPT	This wildcard is used in the form: 'list_1 EXCEPT list_2' This entry matches anything that matches list_1 unless it also matches list_2. A common usage example is 'ALL EXCEPT 123.45.67.89' for host designation or 'ALL EXCEPT in.ftpd' for daemon designations.

Besides these three keywords, TCP Wrapper also allows the usage of domain and IP subnet designation to make configuration easier. To match domains, use a preceding "." prior to the domain. For example '**.**army.mil' will match rtrp.hqisec.army.mil and any other host located in the domain army.mil. To match IP subnets, use an ending "." after the IP subnet area. For example 137.80.99**.** will match 137.80.99.139 and any other host in the 137.80.99 subnet area. If standard subnet masks are not used, an expression of the form 'n.n.n.n/m.m.mm' is interpreted as net/mask pair and can be used as part of the configuration entry.

One of either two basic access control philosophies can be used when establishing your access control logic. In the first method, specific hosts/services are denied while all others are allowed by default. In the second method, specific hosts/services are allowed while all others are denied by default. From a system security perspective the second method implements a stronger security posture. By only allowing connections from hosts/services that you have specifically and consciously allowed, this prevents the intrusion from totally unknown and unforseen hosts that would be allowed by the first method. A simple mechanism to enforce the second method is to use a hosts.deny file that contains the entry

```
      ALL  :    ALL
```

to deny all services to all hosts not explicitly allowed in the hosts.allow file.

Example #2 - Using wildcards and subnets:

```
# /etc/wrap/hosts.allow
#
# Using wildcards, subnets and domain areas
#
ALL EXCEPT in.rlogind, in.fingerd : 137.90.99, .army.mil
in.rlogind                        : 137.90.99.1
in.fingerd                        : LOCAL
#

# /etc/wrap/hosts.deny
#
# Implementation of a blanket deny statement as a safety net.
ALL  :  ALL
#
```

In the above example the hosts.allow file specifies that any host in the subnet area 137.90.99 or the domain area .army.mil will be allowed access to all services on the system except for rlogin and finger. The rlogin service is made available to only the host 137.90.99.1 while the finger service is available to any host on the local network or in the local domain. The hosts.deny establishes a blanket deny statement meaning that all host/services combinations, not allowed by the hosts.allow file, will be denied.

Though two different access control files are normally used, either file can use the keywords *allow* and *deny* to clarify access logic. These keywords instruct TCP Wrapper to accept or reject the connection irregardless of which access control file the line entry is located. The following example shows the usage of these keyword options:

Example #3 - Keywords allow and deny:

```
# /etc/wrap/hosts.allow:
#
# This example shows how the keywords allow and deny can be used to
# clarify and create “exemptions” to a wider access control criteria.
# In this example a single host (137.90.91.139) is the only host
# allowed Telnet access from the 137.90.91 subnet.  Note that the
# larger 137.90 net has been allowed Telnet access.
#
in.telnetd : 137.90.91.139 : allow
in.telnetd : 137.90.91.    : deny
in.telnetd : 137.90.       : allow
ALL        : 137.90.
```

```
ALL        : ALL            : deny
```

In this example only a single access control file (hosts.allow) is used to specify all of the access control logic. The final line of file specifies that all remaining conditions (host/service combinations) be denied access to the system.

4.4.3 Building Initial Access Control Files.

With TCP Wrapper installed and started, it is time to develop the access control logic to support the firewall feature of TCP Wrapper. Determine which hosts will have access to which services on your system. The following are some basic steps that might be used in the development of the initial hosts.allow and hosts.deny files.

Determine system user base. The first things that needs to be done is to sit down and determine what is the actual user base of the system. The user base, and location of systems interfacing with your system, will impact the development of the hosts.allow and hosts.deny files. Try to group the user base into domain or subnet areas according to the services required.

Review TCP Wrapper log file. To get a picture of what machines/services are really accessing your system, the System Administrator should review the entries of the TCP Wrapper log file. How does this compare with the system user base that was developed? Any suspicious or completely unknown hosts gaining access to your system should be researched using *nslookup*, *traceroute*, or other diagnostics utilities that help determine the ownership and location of the host. After this eye-opening adventure, the System Administrator should have a firm grasp of what the actual user base is and which hosts/services need to be denied access.

Carefully create initial hosts.allow and hosts.deny. The operative word in this statement is ***carefully***! When these hosts.allow and hosts.deny are implemented, they will act as a firewall and only allow those hosts/services that you have specifically allowed. A corollary to this is, if you do not get a firm grasp on the requirement (hosts/services) you may either be allowing access that you do not want or denying access to real users with valid requirements to access the system. Real users will get very annoyed if they are suddenly cut-off from the system. As a general rule, start with a fairly simple and generic access control configuration. As time progresses, this configuration can be made increasingly tighter and more specific. To help develop successful access control files, try using the following steps:

1) Use the TCP Wrapper log file as a source of information that you will need to build the initial hosts.allow and hosts.deny files.

2) Build the initial hosts.allow and hosts.deny in a separate area from the actual TCP Wrapper working area. This can be a sub-directory such as /etc/wrap/test/.

3) Test host/service combinations against the trial hosts.allow and hosts.deny using the tcpdmatch utility. Executing the tcpdmatch command with the -d option checks the host/service against the hosts.allow and hosts.deny files in the current directory. The syntax for this action is:

./tcpdmatch -d *service hostname/IP_address*

The tcpdmatch utility will indicated whether this host/service combination would be allowed or denied. When the tcpdmatch utility finds a match in the access control files, it identifies the matched entry for reference. This match can either be an allow action or a deny action.

4) After testing, place the hosts.allow and the hosts.deny file in the TCP Wrapper working area that was designated as part of the Makefile modifications. In our example this directory is /etc/wrap/. Once the files are placed in the working area they will become active. Since TCP Wrapper checks the hosts.allow and hosts.deny upon each connection request, these files are constantly reread and changes appear immediately.

4.4.4 Using Banners.

One of the changes, that was made to the TCP Wrapper Makefile, was to enable the usage of banner statements during the initial connect process. Implementing banners requires that two additional set-up actions be taken.

The first action is the establishment and placement of the banners messages on the system. Banner messages are text files that are named according to the service or daemon that they support. Thus a separate banner message must either be created or a logical link created for each service for which TCP Wrapper will display a banner. The important part of this is that the file name must exactly match the daemon name that is used by the hosts.allow and hosts.deny configuration files. In other words, the banners statement for the Telnet service (in.telnetd) must be located in a file called in.telnetd. However, what if you want to use different banners for the same service depending on whether access has been allowed or denied? Luckily TCP Wrapper supports the usage of multiple directory areas for banner messages. As will be seen in a minute, the hosts.allow and hosts.deny line entries only specify a directory to look into for the appropriate banner message. Thus two different directories could be establish to support banners, one for the accept connections and one for the denied connections. As an example, a Telnet banner for accepted connections could be */etc/wrap/good/in.telnetd* while a Telnet banner for denied connections could be */etc/wrap/bad/in.telnet*.

The second action, required to implement banners, is that the entries in the hosts.allow and hosts.deny need to be modified to add the optional banners statement and specify the location of the banner statement. The line syntax to use with banners is:

daemon_list : client_list [: option] : banners *directory_path*

directory_path is the full path of the directory containing the banner messages

Example from a hosts.allow file:

```
in.telnetd : .army.mil : banners /etc/wrap/good/
```

This entry allows Telnet connections from all army.mil addresses and presents a banner upon the initial connection request. The banner message is located in /etc/wrap/good/ and would be called in.telnetd.

Example #4 - Using Banners:

```
# /etc/wrap/hosts.allow
#
#  This example shows the usage of banner statements.
#
in.telnetd : 137.90., 150.211.90. : banners /etc/wrap/good/
in.ftpd    : .mil except .af.mil  : banners /etc/wrap/good/
#

# /etc/wrap/hosts.deny
#
#  Usage of banners for a denied connection.
#
ALL  : ALL  : banners /etc/wrap/bad/
#
```

This set of example hosts.allow and hosts.deny files shows how two different directories can be used to support different banner messages for accepted and rejected connections. In the hosts.allow file, the first line entry specifies that the Telnet service is available for systems in the 137.90 and 150.211.90 subnets. The banner message for accepted Telnet connections is located in /etc/wrap/good/in.telnetd. The second line entry of the hosts.allow file states that the FTP service is available for all systems in the .mil domain except for those located in the .af.mil domain area. The banner message for accept FTP connections is located in /etc/wrap/good/in.ftpd. In the hosts.deny file all services are denied to all systems. The banner messages for all denied connections is located in the directory /etc/wrap/bad/.

4.5 Advanced Topics Not Addressed.

Due to the limited scope of this paper not all of the features of TCP Wrapper can be adequately explored. Some of these advanced topics and features may be useful in specific situations or on certain systems. The TCP Wrapper README file details these additional topics. The following is a quick summary of some of these features.

Double-reverse. TCP Wrapper can perform a double-reverse lookup of the IP address, making sure that the DNS entries for the IP address and the hostname match.

Ident Protocol. TCP Wrapper uses the ident protocol (RFC 1413) to determine the username associated with the incoming connection. Username information can be used in the access control files to further define access conditions. This requires that the requesting systems support the ident protocol.

Reverse Finger. TCP Wrapper optionally runs a command to get a list of users on a computer that is trying to contract your system.

Jail Environment. Transfers, to a jail or faux environment, are supported to allow observation of suspect systems and users.

Expansions. TCP Wrapper supports shell expansions to allow process execution based on incoming connection data.

4.6 Recommendations.

Because each system is unique and has a unique user base, exact configurations and recommendations can be presented. However, there are a number of simple and basic guidelines that can be used.

1) Unneeded Services. Remove unneeded services from the inetd.conf. In virtually all cases the following services can be removed: echo; discard; chargen; and time. These are old services and are not generally used today. Some other daemons that are not generally used are uucp, comsat, talk, and ntalk.

2) Remote Services. Closely limit remote services such as rlogin, rsh, and rexec. These services use only the hostname as access control logic and can be security vulnerabilities. Through the system hosts.equiv and the users .rhost file, trusted relationships can be created that do not required any password mechanism. TCP Wrapper can be used to prevent widespread abuse of this security vulnerability. Access to these services should be allowed to only specific hosts through the hosts.allow file. The usage of subnet and domain areas may open the system up to unintended utilization of the remote services.

3) TFTP Service. As with the remote services, usage of the TFTP services should be guarded closely. If at all possible, TFTP should be disabled. However, if it is a required service, access to TFTP should be given to specific hosts.

4) Finger. Usage of the network finger service should be limited to the local environment. The information that can be gained from a unrestricted system finger daemon is immense. The information can be leverage to gain access using brute-force attack methods or to perform social engineering attacks to gain additional information.

5) Web Servers. Access to a Web Server can also be limited by using a hosts.allow file to limit those systems that can access the http daemon. In these cases, access limitation can be implemented on either a host basis (very restrictive) or on a subnet or domain basis (mildly restrictive).

6) For Telnet and FTP access, closely monitor the system's user base and adjust the access control parameters accordingly. Some systems may require a wide open operation while other may be able to limit access to local networks or domain areas.

5.0 Simple Watcher (swatch).

Simple Watcher (swatch) is a simple program written in Perl to monitor text files in real time, or after the fact. Swatch allows you to automatically scan log files for particular entries and then take appropriate action.

5.1 Basic Features.

Though there are several facilities and applications that will automatically create logging information, the scanning and interpretation of these files is a time consuming and boring tasks. Wading through pages of log output, looking for items of interest, is one of the down-sides of being a System Administrator. Swatch tries to ease the burden on System Administrators by performing the winnowing of the data chaff to find the pertinent grains of relevant data. Swatch is another set of eyes that can constantly monitor the health of a system by actively scanning log files in real-time. These eyes do not get tired, do not make mistakes, and do not request overtime. Of special importance, critical real-time events can be brought to the attention of the System Administrator or resolved automatically per the System Administrators instructions. Conversely, swatch can be used to review log files to look for particular events that might have happened weeks ago or it can look through the daily logs at night and present the results to System Administrator first thing in the morning.

Swatch keys on phrases, established by the System Administrator in the configuration file, to take specific actions. These actions can be to echo the line to the screen, ring the screen bell, execute another program, pipe the line to another program, mail, or write the line. Each instance of swatch is used to check one file or program output. Multiple instances of swatch can be used to watch over multiple files or program output as desired.

5.2 Installation.

Swatch is not a major system application and only takes a few minutes to install properly. Being a Perl script, Perl must be loaded on the system. Perl is a freely available programming language. Perl, using the best aspects of sed, awk, C, and UNIX shell programming, creates a simple to use, yet powerful, interpretive programming language. Perl is distributed under the standard GNU licensing concept. Getting on with the installation, the swatch utility is available from DISA ASSIST, ACERT and most UNIX software repositories. The current version of swatch is version 2.2.

5.2.1 Installation Steps.

1) Download swatch distribution file. This file, depending on the download location, should be either swatch-2.2.tar.Z or swatch-2.2.tar.gz.

2) Uncompress the swatch file using the appropriate command:

```
uncompress swatch-2.2.tar.Z          OR
gunzip swatch-2.2.tar.gz
```

3) Extract the source files from the swatch archive (swatch-2.2.tar). Note that a swatch subdirectory will be created as part of the extract process.

tar -tvf swatch-2.2.tar	View archive and check for archive integrity
tar -xvf swatch-2.2.tar	Extract contents of the archive file

Change directory to the swatch-2.2 directory:

```
cd swatch-2.2
```

4) Read the swatch README file for pertinent information.

5) Execute the install script using the following commands:

```
sh install.sh
```

6) Answer the questions generated through the swatch installation script. During execution, of the installation script, be prepared to answer the following questions:

a. Enter the directory where you want swatch to be installed.
   Default: /usr/local/etc
   Suggestion: Use a directory area that is already is the normal PATH structure. One suggestion would be /usr/local/bin.

b. What user should own the swatch program files?
   Default: root
   Suggestion: Take default

c. What group should own the installed swatch program files?
   Default: wheel
   Suggestion: Use group normally allocated system administration duties.

d. What should the permissions be for the installed swatch program files?
   Default: 755
   Suggestion: Take default

e. What should the permission be for the installed libraries and man pages?
   Default: 444
   Suggestion: Take default

f. Enter the name of the directory where the Perl libraries are located.
   Default: /usr/local/lib/perl
   Suggestion: Common places for the Perl libraries are as follows:

/usr/lib/
/usr/lib/perl
/usr/lib/perl5
/usr/local/lib
/usr/local/perl
/usr/local/perl5

g. Enter the name of the directory where you wish to install the swatch program files.
Default: Same location as Perl libraries
Suggestion: Take default

h. What directory should the swatch program install the man pages?
Default: /usr/local/man
Suggestion: Take default

i. What should the extension be for the swatch program manual?
Default: 8
Suggestion: Take default

NOTE: If the "man8" directory does not exist under the man directory (see answer to h), the man page will not be placed as required.

j. What should the extension be for the swatch configuration file manual page?
Default: 5
Suggestion: Take default

NOTE: If the "man5" directory does not exist under the man directory (see answer to h), the man page will not be placed as required.

k. Review entries and enter "y" if answers are correct. Entering "n" will allow user to change settings using previous entries as defaults.

7) Go to the directory, where the swatch is installed, and check the executable (swatch) permissions.

5.3 Operating Concepts.
Though swatch is now installed on the system, further explanation is required before things are actually started. The next paragraphs explain the three operating modes that swatch can employ and a few of the option settings that can be used with swatch.

5.3.1 Operating Modes.
Swatch has three different modes of operation. These are a real-time mode, a file mode, and a program output mode. Though each of the modes use a standard configuration file and settings, each mode has specialized operating characteristics that suit the different needs of the System Administrator.

Real-time Mode. In the Real-time mode, swatch monitors your log files in real time looking at lines as they are added to a subject text file. In this mode, swatch is essentially doing a series of "tail" commands to see what has been added to the file and scans these new lines for the pertinent phrases identified in the configuration file. The Real-time mode is very useful in identifying and taking action on those items that are high priority to the System Administrator and may require immediate operator intervention. While in this mode, swatch will be constantly checking the file. Swatch is a "ever-living" job that will consume a session or screen unless run as a background job. The syntax for this mode is:

swatch -t *filename*	*filename* is file being scanned

File Mode. File mode is used to review static or old log files. The Real-time mode constantly looks at the new entries to a file, whereas the File mode performs a single pass through the subject file. When swatch is used in this mode it performs a scan of the file and then exits. This mode is useful in checking logs on a scheduled basis, doing system research using old logs, and copying pertinent information from large log files to smaller more specific files. The syntax for this mode is:

swatch -f *filename*	*filename* is file being scanned

Program Output Mode. This mode differs from the other two in that instead of scanning a file, the standard output of a program is checked for pertinent phrases. For example, this mode can be useful in trapping entries in an application report and system monitoring programs. In addition, this mode can be extremely useful in scanning of information that is not kept in text format, but can be embedded in an application that presents that information in a text format. Of particular applicability is system audit information. Like the Real-time mode, this mode runs as a "ever-living" job that will consume a session, or screen, unless run as a background job. Syntax for the Program Output mode is:

swatch -p *program_name*	*program_name* is program whose output is being scanned

5.3.2 Options.

Swatch has several options settings that affects how swatch operates and how the configuration file is formatted. The two option settings of greatest use are "-c" and the "-r" options.

Configuration File (-c). By default swatch will use the file `.swatchrc` in the users home directory as the configuration file. This can be highly inconvenient for everyday operations so the "-c" option is used to allow the operator to specify the configuration file to be used. This option setting must be used in conjunction with an operating mode setting. Syntax is:

swatch -c *config_file*	*config_file* is new configuration file

Example. The following entry uses the configuration file .sw3rc in the local directory, operates in File mode, and scans the file /var/messages:

```
swatch -c .sw3rc -f /var/messages
```

Restart Time (-r). Using a restart time tells swatch to automatically restart at a specified time or after a period of time. The syntax for this option is:

```
swatch -r restart_time
```

restart_time can be of the forms:	hh:mm[am|pm]	- specific time
	+hh:mm	- after time interval

5.4 Configuration.

Swatch uses configuration files to form the basis of what phrases to trap on and what actions should occur when that phrase is detected. Thus as a minimum, each line entry must contain pattern(s) and action(s). In addition to these two fields, two optional fields can be used to specify how redundant messages are to be handled. Each of the fields being used must be separated by TABs. Each line in the configuration file is a separate entity and does not depend nor is it influenced by any other line in the configuration file. For those who actually use comments, any line beginning with a "#" is ignored by swatch. The syntax for a configuration line entry is:

/pattern/[,/pattern/,…] action[,action,…] [[[HH:]MM:]SS] [start:length]

5.4.1 Patterns.

The first field, /pattern/, specifies a pattern or phrase which is scanned for on each line of the subject file. Multiple patterns are separated using commas with no additional spaces. If more than one pattern is specified, then a match on any of the patterns will signify a match. All patterns are specified as regular expressions. The next few paragraphs lightly touch the basics of using and understanding regular expressions. Additional information can be found in many UNIX system administration, and Perl language, references.

Regular Expression Basics. A regular expression is a pattern – a template – to be matched against a string or in our case a text file line entry. Matching a regular expression against a string either succeeds or fails. If the regular expression is matched against the log file line entry, the associated actions are performed. If the regular expression does not match, then the scan proceeds to the next line. At the absolute fundamental level, a regular expression is a string that is delimited by slashes. Within the slashes, spaces and other whitespace characters are significant just as they are within strings.

/abc/	Matches "abc" exactly
/ab c/	Does not match "abc", but does match "ab c"

Using just simple strings does not accomplish much unless you are looking for exact phrases. To expand the flexibility, regular expressions allow the usage of character classes. These are groups of characters from which one and only one of these characters must be present as part of the subject string

for the pattern to match. Character classes are represented by a pair of open and close square brackets, with a list of characters between the brackets.

/[abcde]/	Matches a string containing any of one of these lower-case characters.
/[a-z]/	Matches a string containing any lower-case alphabetic character.
/[0-9]/	Matches a string containing any number character.
/[dD]enied/	Matches the strings “denied” or “Denied”.

This helps in adding flexibility to the regular expressions, but there is still more. Regular expressions support alteration, as in a|b|c, thereby matching exactly one of the alternatives (a or b or c in this case). Alternation works like an exculsive OR statement.

/a|b|c/	Matches exactly one of the alternatives.
/March|April/	Matches any string containing March or April but not both.

This takes us a step closer, but the best part of regular expressions is the use of wildcards. To begin with a “.” can be used in a regular expression to match any single character except the newline (\n). Effectively this acts like a “do not care” place holder.

/1./	Matches any two character string starting with “1”.
/.at/	Matches any three character string ending with “at”. This would match cat, bat, sat, 7at, etc.

To really become useful we need to combine the “.” with something else. Regular expressions have three multiplier wildcards that will work out perfectly. First, “+” matches one or more of the immediate previous character. Secondly, “?” matches zero or one of the immediate previous character. Lastly, “*” matches zero or more of the immediate previous character. While these multiplier wildcards can be used with standard characters they are more commonly used with the “.”. Of special interest is the “*” (wildcard) combination that essentially matches as characters as required to obtain a match.

/su.*root/	Matches any string containing “su” followed later by “root”. This entry could be used to flag users using the su command in an attempt to become root.
/Nov 12.*warning/	Matches any string (line) containing “Nov 12” followed later by “warning”. This entry could be used to flag warning messages generated on Nov 12.

Through the creative use of these simple regular expression concepts, virtually any type of phrases or combination of phrases can be trapped. Once it is determined what type of events or phrases are to be

trapped by swatch, the usage of these concepts should enable the System Administrator to properly configure swatch to look for these items and take appropriate action.

5.4.2 Actions.

The second field in the swatch configuration file is the action area. Here one or more actions may be specified that will be initiated when a log file line matches the pattern(s) in the first field. As with patterns, multiple actions can be specified. A comma, with no additional spacing, must separate multiple actions. There are seven possible actions that swatch supports. These are echo, bell, exec, ignore, pipe, mail, and write. These are defined below.

echo=[mode]	Echo the matched line. The text mode may be normal (default), bold, underscore, blink, and inverse. Some modes may not work on some terminals.
bell=[N]	Echo the matched line and send bell N times.
exec=command	Execute the command. Variable substitution to command is allowed using fields from the matched line. $N will be replaced with the Nth field, $0 or $* will be replace by the entire line.
ignore	Ignore the matched line.
mail[=address:address:…]	Send mail to address(es) containing the matched line. Default is to user running swatch.
pipe=command	Pipe matched line into command
write[=user:user:…]	User write to send matched lines to user(s)

The variety of actions available should enable the System Administrator to do virtually anything that can be desired when an item is detected. Of particular power is the ability to pipe the trapped line to another program and to initiate another program whenever a particular event occurs.

5.4.3 Redundant Message Timing.

The third and fourth fields of the swatch configuration are optional and are used to control how redundant messages are handled by swatch. The third field, [[[HH:]MM:]SS], is a time interval specified in hours, minutes and seconds. If this interval is specified, and more than one identical line is received, swatch will do the actions specified until the time interval has elapses. This prevents tens to hundreds of identical lines being processed separately. The fourth field, start:length, specifies the location of the first character of the time stamp and the length of the timestamp field. For those log files using timestamps, each line entry will be different due to a different timestamp. This can negate the effect of using the redundant message feature implemented through utilization of the third field. Using the fourth field allows the timestamp to be logically eliminated from the line entries and allows redundant line

entries to be properly handled. This field is only used in conjunction with the third field and needs not be used at all.

Example: /file system full/ echo,bell 01:00 0:16

This example matches a line which contains the string "file system full" and will echo the line on the screen, and sound a bell. Also, multiple instances of message will not be echoed if they appear within a minute of the first one. Instead, the following message will be acted upon after the time interval has expired. This is what may appear if a message appeared 20 times.

```
     ** 20 seen in 00:01:00 == > somehost: /var: file system
full
```

Note that the timestamp has been indicated to start at the first line character and proceeds for a total of 16 characters.

5.4.4 Examples and Explanations.
In this area several example configuration files will be shown and explained to help the System Administrator in the development and implementation of system specific configuration files.

Example 1: The example shows a simple Real-time mode configuration that checks the TCP Wrapper file for refused connections, for connection using the rexec service and for connection originating from the host bholson.hqisec.army.mil. The command line entry to execute this swatch action would look something like the following:

```
swatch -c .sw1rc -t /var/adm/tcpd_log
```

where the configuration file .sw1rc is as follows:

```
#  .sw1rc swatch configuration file
#
#  Capture any refused connections and echo using inverse mode
/refused/               echo=inverse
#  Report usage of rexec service by mailing line entry to
#  sysadmin people
/rexecd/                mail=sysadmin
#  What to track any and all connections from this host.  Echo
#  to screen and ring bell to alert operator.
/bhoson.hqisec.army.mil/     echo,bell
```

For reference here are some sample line entries from a TCP Wrapper log file (tcpd_log).

```
Oct 21 09:05:28 rtrp in.telnetd[301]: connect from bhoson.hqisec.army.mil
Oct 21 09:06:29 rtrp in.rexecd[312]: connect from reedd.hqisec.army.mil
Oct 21 09:19:35 rtrp in.rexecd[370]: connect from bhoson.hqisec.army.mil
```

```
 Oct 21 09:21:06 rtrp in.rexecd[380]: connect from reedd.hqisec.army.mil
 Oct 21 09:22:55 rtrp in.telnetd[392]: connect from reedd.hqisec.army.mil
Oct 21 09:27:01 rtrp in.telnetd[405]: connect from reedd.hqisec.army.mil
Oct 21 09:28:16 rtrp in.telnetd[417]: refused connect from reedd.hqisec.army.mil
Oct 21 09:30:51 rtrp in.telnetd[421]: connect from reedd.hqisec.army.mil
Oct 21 09:42:00 rtrp in.telnetd[448]: connect from reedd.hqisec.army.mil
Oct 21 09:45:02 rtrp in.telnetd[636]: connect from reedd.hqisec.army.mil
Oct 21 10:08:26 rtrp in.ftpd[698]: connect from bhoson.hqisec.army.mil
```

Example 2. By using many of the same attributes and components of the previous examples, a totally different result can be accomplished. By running swatch in the File mode and using different actions for the same search items, swatch can be made act totally differently.

swatch -c .sw2rc -f /var/adm/tcpd_log

```
#  .sw2rc swatch configuration file
#
#  Display using mode=underling any refused connections
#  that occurred since file began.
/refused/               echo=underline
#  Display any connections using the rexecd for visual
#  analysis of what systems are using rexecd.
/rexecd/                echo
#  Copy all entries from the TCP Wrapper log file that
#  contain connections (failed or successful) from the
#  bholson.hqisec.army.mil.  Entries are placed in the file
#  dholson.log and is used to track system usage from that
#  system.
/bholson.hqisec.army.mil/    pipe="cat >> dholson.log"
```

Example 3. In this example, the system's messages file is being examined in real-time for possible system problems. This is done using Real-time mode.

swatch -c .sw3rc -t /var/log/messages

```
#  .sw3rc swatch configuration file
#
#  If file system becomes full, echo with blink to the screen
#  and use call_pager utility to reach operator with code
"1911"
/file system full/ echo=blink,exec="/etc/call_pager 5551234
1911"
#  Log all reboot actions to log file.
/reboot/           echo,pipe="cat >> reboot.log"
#  Want to know if the system starts/restarts with the
#  TFTP server started.
/tftpd/            echo,write=sysadmin
```

Example 4. This examples shows how to use swatch to copy selected lines from an original file (sulog) to other files for archival purposes.

swatch -c .sw4rc -f /var/adm/sulog

```
#  .sw4rc swatch configuration file
#
#  Selected lines from original file and copy to a new file
#  Copy all failed su attempts to root in file failed_root.log
/SU.*:....-.*-root/            echo,pipe="cat >> failed_root.log"
#  Copy all su attempts to root for user dreed to file dreed_root.log
/SU.*dreed-root/               echo,pipe="cat >> dreed_root.log"
#  Copy all su attempt to root for all attempts not originating
#  from the console device.  Line entries are placed in pts_root.log.
/SU.*:.....p.*-root/           echo,pipe="cat >> pts_root.log"
```

For reference here is an excerpt from a sample sulog file.

```
SU 07/30 08:38 + pts/0 dreed-root
SU 10/17 06:45 + console root-dreed
SU 10/21 09:42 + pts/1 dreed-root
SU 10/22 09:46 - console dreed-root
SU 10/22 09:47 + console dreed-root
SU 02/19 14:27 + pts/1 root-cruther
SU 02/24 12:19 + console jcrowley-root
SU 05/09 14:16 + pts/0 cruther-root
SU 05/11 18:29 + pts/0 root-dreed
SU 05/11 18:29 + pts/0 root-dreed
SU 05/11 18:29 - pts/0 root-cruther
SU 05/11 18:40 + pts/2 cruther-root
```

5.5 Implementation.

Swatch is a simple, yet extremely powerful, program. Swatch, with the ability to monitor or review files and then take a series of actions, is a great system administration and security tool. However, to get these benefits from swatch, the System Administrator must know the system inside and out and be able identify the catch phrases that swatch will look for. If the systems administrator does not know what to look for in the various log files, then swatch will not provide any relevant benefits.

To properly implement swatch, a six-step approach should be taken to properly develop and sustain the swatch configuration files.

(1) *Determine files and programs to be monitored.* The System Administrator must spend some time and effort to review the applicable log files, or applications, to determine what files and program output should be monitored by swatch. Swatch should be used for both security monitoring of the TCP Wrapper log and other security and system related information. Check the syslog configuration file for the location of standard system log information generated by syslog.

(2) *Develop a list of phrases to trap on.* A listing of catch phrases should be developed for each file or program. These are the items that the System Administrator should normally be looking for

on a regular basis. Some items may be critical items like " file system full". Other items may only be informational such as a logging of all "su" events.

(3) *Develop action(s) for each catch phrase*. For each catch phrase, an associated action or actions must be developed. Depending on the criticality of the event, you may want to echo the item to screen and execute another program to take action on the event. Perhaps you only want to copy out pertinent entries of the log file into a separate file for ease of administration and research?

(4) *Determine operating mode for each phrase/action*. For each phrase and action pairing, the System Administrator must determine which of the three operating modes should be used. The System Administrator must determine what phrases/actions should be monitored in real-time and which should be performed periodically using File mode. In some circumstances a phrase/action may need to use the Real-time mode for an immediate response and in addition be needed for periodic review for research or data reduction purposes.

(5) *Assign actions/phrases to configuration files*. With a listing of files/program, catch phrases and associated actions in hand, these entries need to be applied to configuration files. Each configuration can only operate in a single mode and can only be applied against a single file or program.

(6) *Review and modify as required*. Periodically review the swatch configuration files for modifications and required changes. Inevitably, the System Administrator will not be able to capture all the needed phrases and actions on the first or even second go around of the configuration files. As system conditions changed, the swatch configuration files should be adjusted.

5.6 Operation.

With the configuration files developed, the operation of swatch is a fairly simple matter. However, a little planning and effort will enable the swatch monitoring to become an automatic part of system operations. Swatch can be used in two different ways, either programmed or on-the-fly.

For constant active monitoring using swatch, the swatch start command should be integrated into the run control (RC) start up sequences. This ensures that each time the system is restarted that swatch will initialize and begin monitoring.

Periodic reviews or data reduction should be integrated into the system using crontab entries to start specific swatch jobs at times convenient to the System Administrator and the system. Review of daily logs or security entries can be performed during off-hours and the output made available for the System Administrator first thing in the morning.

On-the-spot research or investigation may necessitate using swatch in an unscheduled or on-the-fly method. This is accomplished through using the command line with the applicable operating mode and configuration.

5.7 Recommendations.

Though it would be nice to provide exact configurations and guidance on how to configure and operate swatch, it is not really feasible. Each system is different, different in terms of operating system, applications, and configuration. This is the reason why the System Administrator must be keenly aware of what goes on with the system. As part of the implementation guidelines in Section 5.5, general guidance was offered on how to develop appropriate swatch configuration files. Some specific log and text files that should be investigated as part of the swatch configuration process are:

TCP Wrapper logfile
messages files (syslog output file)
su log
syslog file
Applications specific log files
logins log

APPENDIX A
ACRONYMS

ACERT	Army Computer Emergency Response Team
ACT	Access Control Test
ASSIST	Automated Systems Security Incident Support Team
BAT	Binary Authentication Tool
C2	Command and Control
CDT	Change Detection Test
COPS	Computer and Oracle Password System
CQL	Configuration Query Language
DES	data encryption standard
DII	Defense Information Infrastructure
DISA	Defense Information Systems Activity
DISC4	Director of Information Systems for Command, Control, Communications, and Computers
DNS	Domain Name Server
DSN	Defense Switched Network
DSS	Digital Signature Standard
FAQ	frequently asked questions
FTP	file transfer protocol
GUI	graphical user interface
INFOSEC	Information System Security
IP	Internet Protocol
JCDB	Job Control Data Base
JCS	Job Control System
LLNL	Lawrence Livermore National Laboratories
MCS	Master Communications Server
MD5	Master Digest 5
PID	process identifier
PSI	Password Security Inspection

QSP	Quick System Profile
RC	run control
RCS	Remote Communications Server
SPI	Security Profile Inspector
SPI-NET	Security Profile Inspector for Network
swatch	Simple Watcher
TCP	Transmission Control Protocol
UDP	User Datagram Protocol

Appendix B: Additional UNIX C2 Protect Tools

The following paragraphs briefly described additional C2 Protect Tools for UNIX systems. These tools can be found either at DISA, ASSIST, or the ACERT.

Courtney: Is a program that monitors a network and tries to identify an incoming SATAN attack/probe. Courtney receives input from tcpdump, counting the number of new services a machine originates within a specific time period. If one machine connects to numerous services within that window, Courtney identifies the machine as a potential SATAN host.

Message Digest 5: Is a cryptographic checksum UNIX utility program. Message Digest 5 takes message input of varying length and produces a 128 bit fingerprint or message digest number.

Npasswd : Is a replacement for the system password command that incorporates a password checking system that refuses poor password selections. This program reduces the chance of users choosing easily guessed passwords.

Password+: Is a proactive password checker, which replaces the system's password command. Password+ is driven by a configuration file that determines what types of passwords are allowed.

Genpass: Is a program that produces a random account password. Genpass can be used to generate temporary passwords for user's accounts or anywhere else randon password generation is required.

Tripwire: Is a utility that scans a set of designated files and directories, computes a digital signature, then compares the digital signature/fingerprint to a signature previously generated and stored in a database. Differences are flagged and logged including additions and deletions. When used regularly Tripwire enables a System Administrator to spot any changes to files and directories rapidly.

WU-FTPD (Washington University-File Transfer Protocol Daemon): Is a replacement FTP daemon that has security and functional enhancements geared toward anonymous FTP access. **WU-FTPD** replaces the standard FTP server daemons on UNIX systems. Features include extensive logging and a way of limiting the number of FTP users accessing a site.

SMRSH (Sendmail Restricted Shell): Is a sendmail restricted shell utility that provides the ability to specify, through configuration, an explicit list of executable programs. **SMRSH** helps protect against the vulnerability that allows unauthorized remote users to execute programs as any system user. When used in conjunction with sendmail, **SMRSH** effectively limits sendmails scope of program execution to only those programs specified in **SMRSH** 's configuration.

SSH (Secure Shell): Is a program that allows an individual to log into another computer over a network, to execute commands in a remote machine, and to move files from one machine to another. **SSH** is intended as a replacement for rlogin, rsh, rcp and rdist and provides strong authentication and

secure communications over insecure channels.

Portmap : Is Wietse Venema's replacement for the original portmapper program that comes with the operating system. There is an increasing interest in access control for the NIS and RPC-based services that are normally registered with the portmap process. This replacement is intended to reduce the vulnerabilities of the basic program with improved logging and tcp_wrapper like access control lists. This program works with SunOS 4.x releases, Ultrix 4.x, HP-UX 9.x, AIX 3.x and 4.x , and OSF/1. Solaris 2.X and other System V.4 Unixes should use rpcbind.

Rpcbind: Is Wietse Venema's drop-in rpcbind replacement for the standard RPC registry server. This program provides built-in TCP wrapper style client access control and automatically disables proxy access. It provides a simple mechanism to discourage remote access to the NIS, NFS, and other RPC services. Supports SunOS 5.3(Solaris 2.3) earlier Solaris 2.x versions and System V.4 Unix.

Appendix C: The autoReport Feature

While SPI-NET can provide reports with important and pertinent security information. This information is only valid is someone takes the initiative to review the reports and take appropriate actions. Under the standard SPI-NET installation, reports need to be viewed from the command console using the SPI-NET graphical interface. To help ease this process, a UNIX script call autoReport has been developed that will gather unarchived reports from the Incoming Reports list and e-mail the reports to a specified mailbox. This Appendix will provide simple and brief instructions on obtaining, loading and running the autoReport script.

The autoReport script, being affiliated with SPI-NET can be obtained from the same sites and directories as the SPI-NET code itself. Some sample download sites are:

ACERT: ftp.acert.belvoir.army.mil/pub/unix.toolbox/spi/autoReport
ASSIST: ftp.assist.mil/pub/tools/spi-net/autoReport

Installation of autoReport script is a very simple process. The script in distribution form is neither compressed, archived or otherwise altered. For installation the script needs only to be placed in the "binm" directory associated with SPI-NET. This directory would be located at …/spin-1.01/binm/.

Prior to utilizing the script, three variables need to set within the script. These variables are BINMDIR, MAILER and MAILTO.

BINMDIR - Full path to your SPI-NET "binm" directory

MAILER - Any UNIX mailer that accepts "-s subject" on its commandline. "Mail" or "mailx" works on most systems. A full path entry is the best method (e.g., /usr/bin/mail).

MAILTO - Full email address of the intended recipient. The script is designed to only use a single mail address. An alias to an address list can be used to send to multiple recipients.

With these parameters set properly, the autoReport script is ready to run. The autoReport script will automatically collect all the outstanding reports that have been generated since the last time autoReport was run. On an initial installation this may cause many reports to be e-mail to the designated recipient. To prevent this, archive and/or delete unneeded SPI-NET reports. The autoReport script is accessed from the binm directory by the simple command:

```
./autoReport
```

It may be useful to place the binm directory in the execution path through changes to the PATH environment variable. It should be noted that the autoReport script uses the *capitalize* command, and the operator should ensure that this command is available and accessible via the PATH variable.

In order to make the autoReport script even more useful; it should be run as a cron job on regular basis. The interval used depends on your configuration and your operational environment. Obviously, if SPI-NET is scheduled to run jobs with at least a 12-hour interval, it does not accomplish much to run autoReport every hour. Of course there is little downside to running the script frequently since the computational resources required to run the script are minimal. The following line shows a crontab entry to executes autoReport every 6 hours starting at Midnight, every day of the week and month.

```
0 0,6,12,18 * * * /bin/sh /secure/spin-1.01/binm/autoReport 2>&1 > /dev/null
```

For this example, /secure was the source directory for the SPI-NET code. Note that because autoReport is a Bourne shell script, the autoReport command has been prefaced by /bin/sh to force the usage of the required Bourne shell.

For security reasons, this script should be run as part of the root crontab and note that the reports generated and subsequently forwarded to a mailbox should be considered sensitive. Appropriate measures should be taken to secure the root crontab and the mailbox to which the SPI-NET security reports are being sent.